VICTOR DANIEL REYES DREKE

Desenvolvimento de uma ferramenta de validação de
modelos para controladores preditivos baseados em modelos

São Paulo

2020

VICTOR DANIEL REYES DREKE

Desenvolvimento de uma ferramenta de validação de
modelos para controladores preditivos baseados em modelos

Dissertação apresentada à Escola
Politécnica da Universidade de São
Paulo para a obtenção do título de
Mestre em Ciências.

Área de Concentração: Engenharia de
Sistemas

Orientador: Prof. Dr. Claudio Garcia

São Paulo

2020

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catálogo-na-publicação

Dreke, Victor Daniel Reyes

Desenvolvimento de uma ferramenta de validação de modelos para controladores preditivos baseados em modelos / V. D. R. Dreke -- versão corr. -- São Paulo, 2020.
189 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Telecomunicações e Controle.

1. Controle de Processo 2. Controle Preditivo 3. Inteligência Artificial
I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Telecomunicações e Controle II. t.

DEDICATÓRIA

A mi familia

AGRADECIMENTOS

Ao meu querido amigo, orientador e Professor Dr. Claudio Garcia, pela ótima oportunidade, orientação, dedicação, paciência, ensinamentos e apoio que fizeram deste trabalho uma realidade.

A minha família e minha namorada por seu apoio e carinho em todo momento.

Aos meus queridos amigos do LCPI, Manuel Alejandro Pérez Serrano, Hadrien Philippe Martin Desvages, Walter Moreira, Marco Stabilito Mesquita e Edwin Red Estofanero pelo apoio, paciência e ajuda durante a realização deste trabalho.

À comunidade de cubanos da USP, obrigado a todos vocês pelos bons momentos que compartilharam comigo e por estar nos momentos difíceis também.

A CNPq pelo financiamento desse trabalho através de uma bolsa de mestrado.

RESUMO

Quantificar a qualidade de um modelo é um problema existente na área de Identificação de Sistemas. Esta atividade, também conhecida como validação, é fundamental nas aplicações onde se utilizam Controladores Preditivos baseados em Modelos, porque estes precisam de um modelo adequado para seu bom funcionamento. Baseado nisto, nesta dissertação são implementados três algoritmos de Inteligência Artificial capazes de prever, de forma autônoma, quão adequado pode ser um modelo para este tipo de aplicação. Os algoritmos são Árvores de Decisão, Máquina de Suporte de Vetores e Redes Neurais Artificiais. Eles preveem a qualidade do modelo a partir de resultados de outras métricas de validação não existentes. As plantas para a implementação destes algoritmos são: (i) Planta de Clarke (simulada) e (ii) Planta Piloto de Neutralização de pH (real) do Laboratório de Controle de Processos Industriais da Escola Politécnica da Universidade de São Paulo. Em ambos os casos se usa um algoritmo *Dynamic Matrix Control* – DMC ou sua variante *Quadratic Dynamic Matrix Control* – QDMC (em caso de se ter restrições) para executar o controle. Como resultado deste trabalho obtiveram-se algoritmos capazes de prever a qualidade do modelo com uma acurácia de 84,1%, 91,5% e 91,0% para a malha de controle da Planta de Clarke, e de Nível e de pH para a Planta Piloto de Neutralização de pH, respectivamente.

Palavras-chave: Controlador preditivo baseado em modelo; Inteligência Artificial; *Dynamic Matrix Control*; *Quadratic Dynamic Matrix Control*.

ABSTRACT

Quantifying the quality of a model is an existing problem in the Systems Identification area. This task, also known as validation, is fundamental in applications where Model Predictive Control is used, because they need an adequate model for their proper operation. Based on this need, in this dissertation, the author implements three Artificial Intelligence algorithms that are capable of autonomously predicting how suitable a model can be for this type of application. The algorithms are Decision Trees, Support Vector Machine and Artificial Neural Networks. They predict the quality of the model from the results of other non-existent validation metrics. The plants for the implementation of these algorithms are the Clarke Plant (simulated) and the pH Neutralization Pilot Plant (real) of the Industrial Process Control Laboratory of the Polytechnic School of the University of São Paulo. In both cases, a Dynamic Matrix Control - DMC algorithm or its Quadratic Dynamic Matrix Control - QDMC variant (in case of constrained problems) is used to perform the control. This work results are algorithms capable of predicting the model quality with an accuracy of 84.1%, 91.5%, and 91.0% for the Clarke Plant, and the Level and pH control loops pH of the Neutralization Pilot Plant, respectively.

Keywords: Model based predictive controller; Artificial Intelligence; Dynamic Matrix Control; Quadratic Dynamic Matrix Control.

LISTA DE FIGURAS

Figura 3-1- Visão geral dos processos de verificação e validação de modelos.....	15
Figura 3-2- Exemplo de matriz de custos acumulados.	28
Figura 4-1- Diagrama P&ID simplificado da Planta Piloto de Neutralização de pH.	35
Figura 4-2- Diagrama da arquitetura da rede ethernet no LCPI.....	38
Figura 4-3- Fluxograma do processo de identificação.	42
Figura 4-4- Resultado dos experimentos de identificação com degrau na planta piloto. ...	43
Figura 4-5- Resultado dos experimentos de identificação com GBN na planta piloto.	44
Figura 4-6- Comparação entre os dados tratados e reais obtidos na malha de Nível.	45
Figura 4-7- Comparação entre os dados tratados e reais obtidos na malha de pH.....	45
Figura 5-1- Fluxograma do projeto de IA.	48
Figura 5-2- Modelo matemático de um neurônio artificial.	51
Figura 5-3- Gráfico de fluxo de sinais destacando os detalhes do neurônio da saída k conectado ao neurônio oculto j.....	53
Figura 5-4- Arquitetura de uma SVM usando uma rede de base radial.	54
Figura 5-5- Distribuição dos resultados dos testes Hipóteses na planta de Neutralização de pH.....	57
Figura 5-6- Resposta dos modelos No.19 das duas malhas da planta de Neutralização de pH e suas respectivas funções CDF.....	58
Figura 5-7- Resposta dos modelos No. 2 das duas malhas da planta de Neutralização de pH e suas respectivas funções CDF.....	59
Figura 5-8- Distribuição dos resultados dos testes Hipóteses com tempo de amostragem 60s.....	59
Figura 5-9- Comparação das respostas dos modelos No.2045 No.19 No.2949 No.2 e No.747 na malha de Nível	60
Figura 5-10- Comparação das respostas dos modelos No.3044, No.3056, No.3041 e No.3035 da malha de nível.....	61
Figura 5-11- Distribuição dos resultados dos testes Hipóteses na planta de Clarke	62
Figura 5-12- Resposta dos modelos No.16, No.1841, No.1612, No.22 e No.2179 da planta de Clarke e suas respectivas CDF	63
Figura 5-13- Distribuição dos resultados das MAME na planta de Neutralização de pH (a) dados de validação sem processar (b) dados processados	66
Figura 5-14- Respostas dos modelos No.3117, No.1930 e No.1932 da malha de nível ...	66
Figura 5-15- Respostas dos modelos No.1446 e No.1467 da malha de pH.....	67
Figura 5-16- Distribuição dos resultados das MAME na planta Clarke.....	68

Figura 5-17- Respostas dos modelos No.2443, No.4669 e No.4720 da planta Clarke.....	69
Figura 5-18- Distribuição dos resultados da métrica Dvurenceska, DTW e DTW' na planta de Neutralização de pH	70
Figura 5-19- Comportamento dos limias das malhas de controle de nível (a) e pH (b)	71
Figura 5-20- Respostas dos modelos No. 1446 e No. 1467 da malha de pH.....	71
Figura 5-21- Comportamento dos modelos No.19 e No.1948 da malha de pH	72
Figura 5-22- Distribuição dos resultados das métricas VM, DTW e DTW' na planta Clarke	73
Figura 5-23- Comportamento dos limias para a planta de Clarke.....	73
Figura 5-24- Resposta dos modelos No.885, No.2443, No.4669 e No.4720 na planta de Clarke	74
Figura 5-25- Distribuição dos métodos de critérios de seleção de modelos das duas plantas. (a) planta de pH e (b) planta Clarke	75
Figura 5-26- Resposta dos modelos No.3, No.9, No.19, No.25, No.270, No.269, No.1692 e No.1686 da malha de nível.....	76
Figura 5-27- Distribuição dos KPIs da planta de pH	77
Figura 5-28- Resposta da malha de controle de Nível controlada com os MPC usando os modelos No.1296, No.1235 , No.1903. e No.1902.	78
Figura 5-29- Distribuição da resposta transitória da planta de Neutralização de pH (a) e planta Clarke (b)	79
Figura 5-30- Fluxograma para determinar instabilidade.	82
Figura 5-31- Relação entre os resultados de Shapiro-Wilk quando se usa o tempo de amostragem original e o <i>IDNorm</i> na malha de pH.	83
Figura 5-32- Relação entre o índice de FIT e o <i>IDNorm</i> na malha de nível.....	83
Figura 5-33- Relação entre o índice de FIT e o <i>IDNorm</i> na malha de pH.....	84
Figura 5-34- Relação entre a métrica de Dvurenceska e o <i>IDNorm</i> na malha de pH	85
Figura 5-35- Relação entre AIC e o <i>IDNorm</i> na planta Clarke	85
Figura 5-36- Arquitetura da rede neural artificial usada na planta Clarke.....	87
Figura 5-37- Importância estimada das entradas das árvores de decisão para cada uma das malhas de controle.....	87
Figura 5-38- Matriz de Confusão.	88
Figura 5-39- Matrix de Confusão correspondente com a DT da planta Clarke.	90
Figura 5-40- Matrix de Confusão correspondente com a ANN da planta Clarke.	90
Figura 5-41- Matrix de Confusão correspondente com o SVM da planta Clarke.	91
Figura 5-42- Matrix Confusão correspondente com a DT da malha de Nível.	91

Figura 5-43- Matrix de Confusão correspondente com o SVM da malha de Nível.	92
Figura 5-44- Matrix Confusão correspondente com a DT na malha de pH.....	93
Figura 5-45- Matrix Confusão correspondente com a ANN na malha de pH.	93
Figura 5-46- Matrix de Confusão correspondente com o SVM na malha de pH.....	94
Figura 5-47- Resposta ao degrau dos modelos No.4489, No.1636 e No.4469 identificados na planta Clarke.....	95
Figura 5-48- Comportamento da malha de controle da planta Clarke quando o MPC utiliza os modelos No.885, No.2443 e No.4720.....	95
Figura 5-49- Resposta ao degrau dos modelos No.4489, No.1636 e No.4469 identificados na planta Clarke.....	95
Figura 5-50- Comportamento da malha de controle da planta Clarke quando o MPC utiliza os modelos No.4489, No.1636 e No.4469.	96
Figura 6-1- Resposta ao degrau dos modelos No.3117, No.3148, No.19 e No.1930 da malha de Nível.....	99
Figura 6-2- Comportamento da malha de Nível quando MPC emprega os modelos No.19, No.1930, No.3117e No.3148.	99
Figura 6-3- Resposta ao degrau dos modelos No.47, No.158 e No.1932 da malha de Nível.	100
Figura 6-4- Comportamento da malha de Nível quando MPC emprega os modelos No.47, No.158 e No.1932.....	101
Figura 6-5- Resposta ao degrau dos modelos No.962, No.1131, No.1142 e No.1145 da malha de Nível.....	101
Figura 6-6- Comportamento da malha de Nível quando MPC emprega os modelos No.962, No.1131, No.1142 e No.1145.....	102
Figura 6-7- Resposta ao degrau dos modelos No.2350, No.2347, No.2344 e No.2395 da malha de pH.	103
Figura 6-8- Comportamento da malha de pH quando MPC emprega os modelos No.2350, No.2347, No.2344 e No.2395.	103
Figura 6-9- Resposta ao degrau dos modelos No.160, No.2428 e No.2443 da malha de pH.	104
Figura 6-10- Comportamento da malha de pH quando MPC emprega os modelos No.160, No.2428 e No.2443.....	104
Figura 6-11- Resposta ao degrau dos modelos No.11, No.14, No.464 e No.465 da malha de pH.	105

Figura 6-12- Comportamento da malha de pH quando MPC emprega os modelos No.11, No.14, No.464 e No.465.	105
Figura A.1- Representação da Planta de Clarke usando o Simulink®.....	119
Figura A.2- Conteúdo dos blocos principais da planta, “Planta” (a) e “Controlador” (b). .	120
Figura A.3- Resposta da Planta de Clarke em malha aberta.....	120
Figura A.4- Resposta da Planta de Clarke em malha fechada com um degrau unitário no setpoint.	121
Figura A.5- Experimento de identificação em malha fechada usando vários degraus.....	121
Figura A.6- Experimento de identificação em malha fechada usando GBN lento.....	122
Figura A.7- Experimento de identificação em malha fechada usando GBN rápido.	122
Figura A.8- Predição infinitos passos à frente dos modelos identificados.	124
Figura A.9- Diagrama da planta controlada com um MPC.....	125
Figura A.10- Resposta da planta controlada com um MPC com modelo de referência e ruído.	126
Figura A. 11- Resposta da planta controlada com um MPC com modelo de referência sem ruído.	126
Figura B. 1- Experimentos No.1 de Identificação para a planta Neutralização de pH.	129
Figura B. 2- Experimentos No.2 de Identificação para a planta Neutralização de pH.	129
Figura B. 3-Experimentos No.3 de Identificação para a planta Neutralização de pH.	130
Figura B. 4- Experimentos No.4 de Identificação para a planta Neutralização de pH.	130
Figura B. 5- Experimentos No.5 de Identificação para a planta Neutralização de pH.	131
Figura B. 6- Resposta das malhas de controle da planta de Neutralização de pH usando o MPC com modelos ideais.(a) malha de Nível e (b) malha de pH	132
Figura E. 1- Exemplo do gráfico Boxplot.....	144
Figura I.- 1 Representação gráfica da DT da planta Clarke.....	164
Figura I. 2- Representação gráfica da DT da malha de Nível da planta pH.....	164
Figura I. 3- Representação gráfica da DT da malha de pH da planta pH.	165

LISTA DE TABELAS

Tabela 4.1- Lista das variáveis manipuladas.....	35
Tabela 4.2- Lista das variáveis medidas.....	36
Tabela 4.3- Especificações da Planta Piloto de Neutralização de pH.	37
Tabela 4.4- Parâmetros de sintonia dos controladores PID da planta.....	39
Tabela 4.5- Parâmetros de configuração e sintonia do MPC.....	41
Tabela 4.6- Matriz de correlação.	44
Tabela 4.7- Estruturas dos modelos não paramétricos.	46
Tabela 4.8- Resultado dos experimentos de identificação.	46
Tabela 5.1- Estatística dos resultados dos testes de hipóteses da Planta Piloto de Neutralização de pH.	60
Tabela 5.2- Resultado dos testes de hipóteses dos modelos números 2924, 2, 2045, 19 e 747 da malha de nível.	60
Tabela 5.3- Resultado dos testes de hipóteses dos modelos números 3044, 3056, 3041 e 3035 da malha de nível.	61
Tabela 5.4- Estatísticas dos resultados dos testes de hipóteses da planta de Clarke.....	62
Tabela 5.5- Resultados dos testes de hipóteses dos modelos números 16, 1841, 1612, 22 e 2179 da planta de Clarke.....	63
Tabela 5.6- Estatísticas dos resultados das MAME na Planta Piloto de Neutralização de pH com os dados sem processar.	65
Tabela 5.7- Estatísticas dos resultados das MAME na Planta Piloto de Neutralização de pH com os dados processados.....	65
Tabela 5.8- Resultados das MAME dos modelos números 3117, 1930 e 1932 da malha de nível.	67
Tabela 5.9- Resultados das MAME dos modelos números 1446 e 1467 da malha de pH.	67
Tabela 5.10- Estatísticas dos resultados das MAME da planta de Clarke.....	68
Tabela 5.11- Resultados das MAME dos modelos números 2443, 4669 e 4720 da planta de Clarke.	69
Tabela 5.12- Estatísticas da distribuição dos resultados das métricas VM, DTW e DTW'. 70	
Tabela 5.13- Resultados das métricas VM, DTW e DTW' dos modelos números 1446, 1467, 19 e 1948 da malha de pH.....	72
Tabela 5.14- Estatística dos resultados das métricas VM, DTW e DTW' na planta de Clarke.	73

Tabela 5.15- Resultados da validação com as métricas VM, DTW e DTW' dos modelos números 885, 2443, 4669 e 4720 da planta de Clarke.	74
Tabela 5.16- Estatísticas dos KPIs dos experimentos número 3 das duas plantas.	75
Tabela 5.17- Resultados dos critérios de seleção dos modelos números 3, 9, 19, 25, 270, 269, 1692 e 1686 da malha nível.	77
Tabela 5.18- KPIs da malha de controle de nível controlada com os MPC usando os modelos números 1296, 1235, 1903 e 1902.	78
Tabela 5.19- Estatísticas das respostas transitórias das duas plantas.	80
Tabela 5.20- Declaração das classes.	81
Tabela 5.21- Características dos algoritmos de IA treinados.	86
Tabela 5.22- Resultados da AUC das classes dos algoritmos de IA treinados.	89
Tabela 5.23- Comparação entre os resultados de validação e as classes preditas dos modelos números 885, 2443, 4720, 4489, 1636 e 4469 da planta de Clarke.	94
Tabela 5.24- Comparação entre os resultados dos KPIs quando são empregados os modelos números 885, 2443, 4720, 4489, 1636 e 4469 da planta de Clarke.	96
Tabela 5.25- Polinômios dos modelos usados nas Figura 5-47 e Figura 5-49.	96
Tabela 6.1- Resumo dos resultados dos modelos testados na Planta Piloto de Neutralização de pH.	98
Tabela A. 1 Características dos modelos obtidos no Teste nº 1.	123
Tabela A.2- Informações sobre os modelos da Figura A.8.	124
Tabela A.3- KPIs de referência da Planta de Clarke.	127
Tabela B. 1- Características dos Experimentos do Teste No 1 na planta de Neutralização.	128
Tabela B. 2- KPI das malhas de controle da planta de Neutralização de pH usando o MPC com modelos originais.	132
Tabela B. 3- Valores de referências das distintas malhas de controle para normalizar os KPIs.	132
Tabela C. 1- Estatísticas dos KPI normalizados das classes catalogadas.	133

LISTA DE ABREVIATURAS E SIGLAS

AIC	<i>Akaike Information Criterion</i>
ANN	<i>Artificial Neural Network</i>
AUC	<i>Area Under the Curve</i>
BIC	<i>Bayesian Information Criterion</i>
BMI	<i>Bilinear Matrix Inequality</i>
CART	<i>Classification and Regression Trees</i>
CDF	<i>Cumulative Distribution Function</i>
CHAID	<i>Chi-Squared Automatic Interaction Detection</i>
CLP	Controladores Lógicos Programáveis
CP	Coeficiente de Performance
DMC	<i>Dynamics Matrix Control</i>
DT	<i>Decision Tree</i>
DTW	<i>Dynamics Time Warping</i>
EF	Eficiência do Modelo
EPUSP	Escola Politécnica da Universidade de São Paulo
e_{ss}	Erro de regime estacionário
FN	<i>False Negative</i>
FP	<i>False Positive</i>
FPE	<i>Final Prediction Error</i>
FSR	<i>Finite Step Response</i>
GBN	<i>Generalized Binary Noise</i>
IA	Inteligência artificial
IAE	<i>Integrated Absolute Error</i>
IAU	<i>Integral of Absolute Control Action</i>

ITAE	<i>Integral of Time and Absolute Error</i>
ITSE	<i>Integral of Time and Squared Error</i>
KPI	<i>Key Performance Indicators</i>
LCPI	Laboratório de Controle de Processos Industriais
LMI	<i>Linear Matrix Inequality</i>
MAME	<i>Métricas Adimensionais da Média do Erro</i>
MESR	Média do Erro relativo à saída do processo com respeito à referência
ML	<i>Machine Learning</i>
MMS	<i>Manufacturing Message Specification</i>
Mp	<i>Maximum peak</i>
MPC	<i>Model Predictive Control</i>
MRI	<i>MPC Relevant Identification</i>
MRSR	Média da Razão entre a diferença dos valores consecutivos da saída e a referência
MSE	<i>Mean Squared Error</i>
MV	Variável manipulada
P&ID	<i>Piping and Instrumentation Diagram</i>
PI	Proporcional – Integral
PID	Proporcional – Integral – Derivativo
PPV	<i>Positive Predictive Value</i>
PV	Variável controlada
PWM	<i>Pulse Width Modulation</i>
QDMC	<i>Quadratic Dynamic Matrix Control</i>
QUEST	<i>Quick, Unbiased, Efficient, Statistical Tree</i>
RME	<i>Relative Magnitude Error</i>
RMSPE	<i>Root Mean Squared Percent Error</i>

ROC	<i>Receiver Operating Characteristic</i>
SDCD	Sistema Digital de Controle Distribuído
SP	<i>Setpoint</i>
SVM	<i>Support Vector Machine</i>
TAP	Tanque de Ácido Primário
TAPI	Tanque de Ácido Primário Intermediário
TAS	Tanque de Ácido Secundário
TASI	Tanque de Ácido Secundário Intermediário
TBB	Tanque de Base da Bomba
T_D	Tempo Derivativo
TF	<i>Filter Time Constant</i>
T_I	Tempo Integral
TIC	<i>Theil Inequality Coefficient</i>
TN	<i>True Negative</i>
TNR	<i>True Negative Rate</i>
TP	<i>True Positive</i>
TPR	<i>True Positive Rate</i>
TR	Tanque Reator
t_r	<i>Rise time</i>
t_s	<i>Settling time</i>

LISTA DE SÍMBOLOS

AD	Estatística de Anderson-Darling
A	Nível de significância
a	Constantes geradas pela média
b_k	Bias do neurônio k
β	Beta factor (<i>setpoint factor</i>)
$C_r, C_{S\&G}$	Erro global em função do erro relativo de magnitude e fase das métricas de Russell, e de Sprague e Geers, respectivamente
d, d'	Índice de concordância de Willmott
dw	Função de custo do método DTW
D	Estatística de Kolmogorov-Smirnov
Dx_i	Vetor com as diferenças de duas amostras
$e_A(t)$	Erro absoluto
e_k	Erro relativo normalizado
e_{th}	Umbral do erro
$\bar{e}$	Média de erro absoluto normalizado
$\varepsilon(t, \hat{\theta}_N)$	Vetor com os erros de predição
$F_0(x), F(x)$	Função de distribuição acumulativa contínua hipotética
G_c	Ganho do controlador PID da ABB
$\mathbf{G}$	Matriz dinâmica do sistema
g_i	Coefficientes da resposta ao degrau
ξ_k	Sequência de amostras residuais independentes e identicamente distribuídas
$f_m(x)$	Função discriminante do nó
$f(t + k)$	Resposta livre do sistema

H_0, H_1	Hipóteses nula e alternativa, respectivamente
I_k	Parte imaginária do erro aplicando a transformada discreta de Fourier
J	Função custo
$K(x^t, x)$	Função Kernel
M	Valores ordenados esperados do normal padrão
M_{HS}	Erro associado à magnitude da métrica de Sarin
$M_r, M_{S\&G}$	Magnitude do erro relativo das métricas de Russell e de Sprague e Geers, respectivamente
$M_{n,k}$	Ganho normalizado do erro aplicando a transformada discreta de Fourier
η_{Harris}	Índice de Harris
η	<i>Learning rate</i>
n, N	Quantidade de elementos dos dados
p	Probabilidade
P	Precisão
P_{HS}	Erro associado à fase da métrica de Sarin
$P_r, P_{S\&G}$	Fase do erro relativo das métricas de Russell e de Sprague e Geers
R	<i>Recall</i>
R_k	Parte real do erro aplicando a transformada discreta do Fourier
r_{Wi}	Índice <i>Watchdog</i>
$r(t)$	Referência (<i>setpoint</i>)
T_A	Índice da estatística ANOVA
T	Período de amostragem
T_{HS}	Erro associado à topologia da métrica de Sarin
t_{start}	Tempo de início do sinal

$S_n(x), F_n(x)$	Função de distribuição cumulativa contínua observada
$u_{cal}, u_{deco}, u_{exp}$	Incerteza de medição, de modelagem e total
u_k	Combinação linear do neurônio k
$u(t)$	Esforço de controle, variável manipulada
μ_{R_k}, μ_{I_k}	Média da parte real e imaginária do erro
μ_{Wx}	Média usada na estatística de Wilcoxon
μ	Média de um vetor
V	Matriz de covariância
VM	Índice da métrica de Dvurecenska
W	Estatística de Shapiro-Wilk
w_e	Peso de cada erro relativo normalizado
w_{ki}	Peso da entrada i do neurônio k
Wx	Somatória dos <i>rankings</i> usados na estatística de Wilcoxon
$\bar{x}$	Média dos dados observados
$\mathbf{x}$	Vetor das entradas
x_i	Entrada i
χ^2	Distribuição Chi-Quadrado
$\hat{y}$	Valores ou saídas preditas ou simuladas
y	Valores ou saídas medidas ou observadas
y_k	Saída do neurônio k
$y_i^{ts}, \hat{y}_j^{ts}$	Valores ou saídas reamostradas, observadas e preditas
Z_{stat}	Estatística de Wilcoxon
Ω_0	Frequência fundamental
σ_c	Desvio padrão do erro absoluto normalizado

σ_{cap}	Capacidade do desvio padrão do índice de Harris
σ_{tot}	Desvio padrão total da saída
$\sigma_{MV'}$	Desvio mínimo teórico
σ_{Wx}	Desvio padrão usado na estatística de Wilcoxon
σ	Desvio padrão de um vetor
σ^2_{MV}	Variância mínima teórica
σ^2_y	Variância atual da saída do processo
$\sigma^2_{R_k}, \sigma^2_{I_k}$	Variância da parte real e imaginária do erro
Φ	Função de medição das impurezas
φ	Função de ativação
δ_j e δ_k	Gradientes locais dos neurônios k e j
λ	Constante de momento
Y	Pesos da saída do processo
ϑ	Pesos da variável manipulada
H	Matriz hessiana
Γ	Matriz diagonal dos pesos da saída
Λ	Matriz diagonal dos pesos da entrada
δ	Vetor do gradiente
I_p, I_m	Matriz identidade de dimensão p e m , respectivamente
Ψ	Matriz de custos acumulados

SUMÁRIO

1	INTRODUÇÃO	1
1.1	Motivação.....	3
1.2	Objetivos.....	4
1.3	Estrutura do trabalho.....	5
2	REVISÃO DA LITERATURA.....	7
2.1	Literatura relativa aos métodos de validação de modelos.....	7
2.2	Literatura referente à inteligência artificial.....	9
2.3	Literatura referente ao controle preditivo baseado em modelos	11
3	MÉTODOS DE VALIDAÇÃO	14
3.1	Validação de modelos matemáticos.....	14
3.1.1	Métodos Quantitativos	15
3.1.1.1	Teste de hipóteses	16
3.1.1.1.1	TESTE DE KOLMOGOROV-SMIRNOV (K-S)	17
3.1.1.1.2	TESTE DE ANDERSON-DARLING (A-D).....	18
3.1.1.1.3	TESTE DE SHAPIRO-WILK (S-W)	18
3.1.1.1.4	TESTE DE WILCOXON (W_x).....	19
3.1.1.2	Métricas Adimensionais da Média do Erro	21
3.1.1.2.1	EFICIÊNCIA DO MODELO.....	21
3.1.1.2.2	MÉTRICA DE THEIL.....	22
3.1.1.2.3	MÉTRICA DE WILMOTT	22
3.1.1.2.4	MÉTRICA DE RUSSELL.....	23
3.1.1.2.5	MÉTRICA DE SPRAGUE E GEERS.....	24
3.1.1.2.6	MÉTRICA ANOVA.....	25
3.1.1.3	Métrica de Dvurecenska	26
3.1.1.4	Métrica da Deformação Dinâmica do Tempo (DTW)	27
3.1.1.5	CrITÉRIOS de seleção de modelos	29
3.1.1.6	Validação da dinâmica do modelo	30
3.2	Avaliação das malhas de controle	30
3.2.1	Métricas estatísticas	31
3.2.2	Índice de Harris	32
3.2.3	Índice Watchdog	33
3.2.4	Integral Absoluta da Ação de Controle.....	33
4	TRABALHO NA PLANTA PILOTO DE NEUTRALIZAÇÃO DE pH.....	34
4.1	Descrição da Planta Piloto de Neutralização de pH do LCPI.....	34
4.1.1	Malhas de controle	35
4.1.1.1	Controle de pH no Tanque Reator TR	36

4.1.1.2	Controle de nível no Tanque Reator TR	36
4.1.2	Especificações da Planta Piloto de Neutralização de pH	37
4.1.3	Métodos de controle	37
4.1.3.1	Controladores PID	38
4.1.3.2	Controlador MPC	39
4.2	Projeto de Identificação	42
5	PROJETO DE INTELIGÊNCIA ARTIFICIAL	47
5.1	Métodos de Aprendizagem Automática	47
5.1.1	Árvores de Decisão	49
5.1.2	Redes Neurais Artificiais	51
5.1.3	Máquina de Vetores de Suporte	53
5.2	Etapa de obtenção e preparação dos dados para o treinamento	55
5.2.1	Análise dos dados de treinamento	56
5.2.1.1	Análise dos Testes de Hipóteses	57
5.2.1.2	Análises das Métricas Adimensionais da Média do Erro	64
5.2.1.3	Análises das Métricas de Limiar e DTW	70
5.2.1.4	Análises dos Critérios de Seleção de Modelos	75
5.2.1.5	Análise dos KPI na etapa de simulação	77
5.2.2	Definição das classes	80
5.3	Etapa de treinamento	82
5.4	Etapa de validação	88
6	PERFORMANCE DOS ALGORITMOS DE IA NA PLANTA DE pH	97
7	CONCLUSÕES E TRABALHOS FUTUROS	106
7.1	Propostas de trabalhos futuros	109
	REFERÊNCIAS BIBLIOGRÁFICAS	111
	APÊNDICE A – Planta de Clarke	119
	APÊNDICE B – Simulador da Planta Piloto de Neutralização de pH	128
	APÊNDICE C – Distribuição dos KPIs nas classes de treinamento	133
	APÊNDICE D – Histogramas entre as entradas e o <i>IDNorm</i>	134
	APÊNDICE E – Métodos Qualitativos de Validação	144
	APÊNDICE F – Script de Matlab® para treinamento das Árvores de Decisão	146
	APÊNDICE G – Script de Matlab® para treinamento das Redes Neurais Artificiais	153
	APÊNDICE H – Script de Matlab® para treinamento das Máquina de Vetores de Suporte	157
	APÊNDICE I – Representação gráfica das Árvores de Decisão treinadas	164
	APÊNDICE J – Tabela com os modelos testados na Planta Piloto de Neutralização de pH	166

1 INTRODUÇÃO

O controle automático de processos é uma área da Engenharia que influenciou o desenvolvimento da indústria moderna. Seu objetivo é “que o produto fabricado siga certas especificações, baseadas em critérios de qualidade e uniformidade” (GARCIA, 2017). Há o interesse de controlar os processos desde o início das grandes civilizações. Mas não foi até o início da Revolução Industrial que a medição local e o controle manual iniciaram o percurso que faria a Automação e o Controle assumirem a importância que têm hoje. Os sistemas de controle têm evoluído muito, desde o controle local com os controladores caixa-grande da década de 1930 até os Controladores Lógicos Programáveis (CLP) e os Sistemas Digitais de Controle Distribuído (SDCD) (GARCIA, 2017).

O surgimento de novas tecnologias combinadas com os altos níveis de demanda e de competitividade do setor produtivo influenciaram bastante o desenvolvimento de novas técnicas de controle mais precisas e eficientes. Estes fatores são considerados os principais responsáveis pela evolução do controle automático de processos. Atualmente, não é suficiente controlar só de forma estável o processo, a otimização deve ser considerada durante o projeto dos sistemas de controle em muitas indústrias (XI; LI; LIN, 2013).

Entre as técnicas usadas para atingir essas exigências atuais está o controle preditivo baseado em modelo (MPC), o qual tem como característica principal o uso de um modelo dinâmico do processo para prever a variável controlada em um horizonte futuro e calcular a ação de controle que minimize uma função custo (HONC *et al.*, 2016). O MPC é aplicado há mais de 30 anos. Durante esse período, ele foi bem sucedido e foi reconhecido como uma boa solução para lidar com problemas de otimização altamente complexos, nos quais as restrições das variáveis devem ser consideradas (DARBY; NIKOLAOU, 2012; XI; LI; LIN, 2014; FORBES *et al.*, 2015).

Mesmo assim, nas publicações sobre o estado da arte dessa técnica de controle, é comum citar que há muitas coisas a melhorar. O número de casos de sucesso é desigual, algumas empresas obtêm sucesso e outras não. Este fenômeno é atribuído a diferentes causas e trabalhos como os de (DARBY; NIKOLAOU, 2012; FORBES *et al.*, 2015) citam algumas das mais frequentes. Elas podem ser agrupadas dependendo da fase do projeto de controle da seguinte maneira:

- Causas de erros na fase de identificação do modelo:

- Projeto inadequado de experimentos de identificação;
- Uso de métodos de identificação incorretos; e
- Escolha incorreta do modelo ou do seu tipo de estrutura.
- Causas de erros na fase de sintonia do controlador:
 - Complexidade dos métodos de sintonia;
 - Ausência de técnicas determinísticas confiáveis para sintonia; e
 - Dificuldade em lidar com modelos não lineares.
- Causas de erros na fase de operação:
 - Falta de mecanismo de detecção e diagnóstico de falhas para estes controladores;
 - Inexistência ou mau funcionamento de procedimentos para atualização de modelos; e
 - Dificuldade em lidar com plantas não lineares e mal condicionadas.

Em muitas destas etapas, o ser humano desempenha um papel importante. É por isso que a presença de pessoal treinado em cada uma delas é muito importante para o sucesso destes sistemas. A ausência desses profissionais também é causa de erro que afeta todas as etapas. Uma alternativa para este problema poderia ser converter o comissionamento desses controladores em um processo automatizado e intuitivo para a maioria das pessoas (FORBES *et al.*, 2015).

De todos os problemas supramencionados, esta dissertação vai estar focada na escolha correta dos modelos a serem usados neste tipo de controle. Para isto, parte-se da seguinte premissa: em uma aplicação de controle na qual é usado um MPC que não varia seus parâmetros de configuração e sintonia, o modelo mais adequado para ser usado é aquele que produz um melhor comportamento na malha de controle. Usando essa premissa como critério de validação, a dissertação tentará responder a seguinte pergunta: É possível implementar um algoritmo de Inteligência Artificial capaz de validar de forma automática e confiável o modelo a ser utilizado em um MPC?

Este trabalho está dividido em três partes fundamentais. A primeira parte aborda os diferentes métodos de validação usados em situações gerais. A segunda parte apresenta e implementa os algoritmos de aprendizagem automática usados para resolver o problema de validação. A terceira parte é a implementação destes algoritmos na Planta Piloto de Neutralização de pH, localizada no Laboratório de Controle de Processos Industriais (LCPI) da Escola Politécnica da Universidade de São Paulo (EPUSP).

1.1 Motivação

Baseado nas ideias expressas por (FORBES *et al.*, 2015) e (MORALES ALVARADO, 2017), desenvolveu-se um método de validação de modelos lineares para MPC usando um sistema difuso (*fuzzy*) como ferramenta principal. Este tem duas etapas fundamentais: a primeira é coletar informações de diferentes métodos de validação, como: métodos de validação estatística, validações dinâmicas, análise de robustez e métricas estatísticas com base no erro; a segunda usa essas informações como entradas para um sistema de inferência *fuzzy*, que usa regras baseadas no conhecimento do autor para retornar uma classificação quantitativa da qualidade do modelo. Os resultados neste trabalho são díspares. Por um lado, em uma malha com alta não linearidade foi obtido 50% de sucesso; enquanto em uma malha com baixa não linearidade obteve-se 85,71% de sucesso. Apesar disso, o trabalho de (MORALES ALVARADO, 2017) é catalogado como uma boa maneira de abordar esse tipo de validação de modelo.

O uso da Inteligência Artificial (IA) em tarefas de validação de modelos para controladores preditivos é listada como uma das contribuições de (MORALES ALVARADO, 2017). Este não é o único exemplo de trabalho que usa técnicas de IA em tarefas de validação de modelo. Pode-se citar como outros casos as publicações de (SORNETTE *et al.*, 2007) e (SARIN *et al.*, 2009), embora nenhum deles tenha se concentrado em modelos usados em MPC. Os resultados destas publicações também foram positivos utilizando técnicas *fuzzy* recursivas e regressão linear, respectivamente.

A IA é uma área da ciência da computação que estuda algoritmos que respondem e funcionam como seres humanos. Atualmente, é uma das áreas de estudo e inovação em que há mais pessoas envolvidas e mais resultados foram alcançados. O desenvolvimento de computadores mais rápidos e baratos, o aumento da quantidade e do fluxo de dados, juntamente com a otimização dos algoritmos de aprendizado, são fatores decisivos na proliferação deste ramo do conhecimento. Desde seu surgimento nos anos 1950, os algoritmos de IA vêm evoluindo e alcançando resultados como: derrotar humanos em jogos de raciocínio e esportes, identificar rostos e imagens, dirigir carros, reconhecer vozes e ordens, prever o clima etc. Estes exemplos demonstram a capacidade dos métodos de IA de enfrentar e resolver os problemas de aprendizado mais distintos e complexos.

Após o exposto, pode-se afirmar o seguinte como motivações deste trabalho: a falta de métodos de validação de modelos para MPC; a existência de vários exemplos que

mostram bons resultados no uso da IA em tarefas de validação; a melhoria da capacidade de aprendizado e precisão destes algoritmos; a possibilidade de melhorar os resultados de (MORALES ALVARADO, 2017) usando outros métodos de IA e a possibilidade de desenvolver uma nova ferramenta de validação.

1.2 Objetivos

O objetivo principal deste trabalho é desenvolver uma ferramenta métrica cuja função seja validar, invalidar e selecionar modelos a serem usados em controladores MPC. Os algoritmos de IA são a base para gerar esta ferramenta de validação, que são treinados e validados em dois ambientes: na Planta de Clarke e no simulador da Planta Piloto de Neutralização de pH e na planta real.

A partir deste objetivo geral, o trabalho é dividido em objetivos específicos e suas respectivas tarefas, que são apresentados a seguir:

- Obter um conjunto de dados para treinar os algoritmos de IA: uma série de métodos de validação e índices de desempenho (KPIs) é escolhida e; a partir das respostas obtidas, são geradas as entradas e saídas para a ferramenta de validação, respectivamente. As tarefas envolvidas nesta fase são:
 - Escolher e implementar os métodos de validação e os KPIs que melhor possam servir para a finalidade deste trabalho;
 - Projetar e realizar experimentos de identificação usando a Planta de Clarke (vide Apêndice A) e o simulador da Planta Piloto de Neutralização de pH (Apêndice B);
 - Gerar e validar um conjunto de modelos não paramétricos que descrevam os experimentos anteriores; e
 - Avaliar, segundo os resultados dos KPIs, o desempenho do MPC usando estes modelos validados.
- Desenvolver algoritmos de IA para validar os modelos obtidos: três métodos foram escolhidos para tentar resolver o problema descrito. Neste caso, os resultados da validação dos modelos serão usados como entrada e os da avaliação da malha de controle como saída. As tarefas envolvidas nesta fase são:
 - Escolher três métodos que melhor possam atender ao objetivo geral;
 - Treinar os algoritmos selecionados usando o Matlab®; e
 - Escolher o algoritmo de IA que tenha maior precisão.

- Implementar a ferramenta de validação na Planta Piloto de Neutralização de pH do LCPI: usando os resultados do objetivo anterior, o algoritmo de IA é usado para validar os modelos identificados, que representam as malhas de pH e de nível desta planta. As tarefas envolvidas nesta fase são:
 - Realizar experimentos de identificação de malha fechada nas malhas de controle mencionadas;
 - Obter um conjunto de modelos usando o *software* de identificação CASI da *startup* Cursor Identificação e Controle; e
 - Comissionar um controlador MPC do tipo *Quadratic Dynamic Matrix Control* - QDMC (desenvolvido usando Matlab®) nas malhas determinadas.

Os seguintes resultados são esperados:

- ✓ Ter um algoritmo treinado, capaz de fornecer uma qualificação quantitativa sobre a validade do modelo;
- ✓ Ter o conjunto de modelos que descrevam as malhas da Planta Piloto de Neutralização de pH que interessam; e
- ✓ Ter um controlador MPC do tipo QDMC em execução na planta.

1.3 Estrutura do trabalho

No Capítulo 1, Introdução, é apresentado o contexto em que esta dissertação é desenvolvida. É dividido em seções que explicam as motivações do trabalho, seus objetivos e a estrutura do texto.

No Capítulo 2, denominado Revisão da Literatura, são expostos os resultados das pesquisas sobre o estado da arte dos três principais temas que se enquadram neste trabalho: métodos de validação, IA e controle de processos usando MPC.

No Capítulo 3, chamado de Métodos de Validação, os métodos de validação usados são descritos. Este capítulo está dividido em duas partes principais: métodos de validação de modelo e métodos de avaliação de malhas de controle. Para todos os elementos que compõem as duas partes, são explicadas algumas de suas características, a maneira de interpretar seus resultados e as fórmulas para implementá-los.

No Capítulo 4, intitulado Trabalho na Planta Piloto de Neutralização de pH, ela é descrita sucintamente. Como parte desta descrição, ênfase é colocada no sistema de

controle que é usado, tanto a parte teórica quanto a prática são abordadas aqui. Por fim, são relatados os experimentos de identificação realizados e seus resultados.

No Capítulo 5, denominado Projeto de IA, é explicada a base teórica dos algoritmos utilizados. Em seguida, são descritas as etapas de preparação dos dados, treinamento e validação que envolvem a realização desse tipo de projeto. Os resultados mostrados neste capítulo são o produto do trabalho nos ambientes simulados, porém a descrição destes ambientes é feita nos Apêndice A e B.

No Capítulo 6, chamado de Performance dos Algoritmo de IA na Planta Piloto de Neutralização de pH, são apresentados os resultados dos experimentos de controle na planta piloto e são comparados estes resultados com os preditos pelos algoritmos de IA.

Finalmente, no Capítulo 7 são apresentadas as conclusões e as propostas para trabalhos futuros.

2 REVISÃO DA LITERATURA

Este capítulo lista os resultados da pesquisa relativos às áreas de interesse para esta dissertação, tendo como suporte teses, artigos e livros relacionados ao tema. Entretanto, os detalhes da literatura são citados, quando necessário, nos capítulos seguintes. A revisão da literatura foi dividida nos seguintes tópicos: validação de modelos, IA e MPC.

2.1 Literatura relativa aos métodos de validação de modelos

As técnicas para validar modelos são variadas e de natureza distinta. Dada essa variedade, decidiu-se examiná-las em grupos compostos por métricas baseadas em princípios similares ou idênticos. Um desses grupos foram os testes não paramétricos (métricas baseadas em testes de hipóteses). Relacionado a estas técnicas, foram analisados trabalhos como os de Leotti Torman, Coster & Riboldi (2012), no qual foi feita uma revisão e comparação dos principais testes não paramétricos. A ideia é obter um melhor conhecimento da capacidade de verificar a normalidade dos dados de diferentes testes. Dentre seus resultados, destacou-se a superioridade dos testes Shapiro-Francia e Shapiro-Wilk; em contrapartida, verificou-se que os testes de Kolmogorov-Smirnov e Qui-Quadrado demonstraram desempenho inferior.

Sheskin (2000) foi um dos livros usados para entender e estudar esses testes não paramétricos. Todos os métodos usados neste trabalho são discutidos nesta revisão, além disto, foi acrescentado como material adicional tabelas estatísticas e exercícios com suas respostas sobre estes tópicos. Outro livro com características similares usado como referência para o estudo de tais testes é o de Berenson, Levine & Krehbiel (2012). Em seu conteúdo há explicações e metodologias dos testes de Kolmogorov-Smirnov, Anderson-Darling, Shapiro-Wilk, Shapiro-Francia e do teste de Wilcoxon. Além disto, aparecem também as tabelas das diferentes distribuições estatísticas.

Para se ter uma ideia do estado da arte desses métodos, em quais áreas e como são usados, outros trabalhos foram analisados. Nestes, os resultados da implantação dessas técnicas foram satisfatórios. Pode-se destacar: Naghili *et al.* (2013) que as usa para determinar a precisão e fidelidade de dois modelos que descrevem técnicas de alocação de bactérias para cultivo; Luengo, García e Herrera (2009) que as usa para

validar o desempenho de redes neurais artificiais (ANN) e Máquina de Vetores de Suporte (SVM) em tarefas de classificação; Kleijen (1999) demonstra quais testes estatísticos não paramétricos podem ser usados de acordo com os dados disponíveis em diferentes situações.

Entre os trabalhos vistos, houve uma tendência de propor métricas baseadas na conjunção de várias técnicas conhecidas, as quais exibiram resultados satisfatórios. Neste trabalho, quatro delas foram de interesse: Sornette *et al.* (2007a), Sarin *et al.* (2009), Mongiardini *et al.* (2013) e Dvurecenska *et al.* (2018).

Sornette *et al.* (2007a) formularam um algoritmo de validação de modelos como um processo construtivo iterativo, que tenta reproduzir o que normalmente ocorre na mente dos cientistas. Neste caso, são substituídas as afirmações estáticas sobre a impossibilidade de validar um modelo usando um processo dinâmico de aproximação, porém este método se adapta melhor à natureza *fuzzy* da validação. Nesta dissertação se ilustrou a metodologia com vários sistemas reais na área da Física.

Sarin *et al.* (2009) propuseram uma nova métrica para quantificar as discrepâncias entre as respostas dos sistemas de segurança veicular, em que os autores sugerem classificar os componentes do erro associados com as três principais características da resposta (fase, magnitude e topologia). Para tal, esta nova métrica foi concebida baseando-se em outros métodos de análise de dados já existentes. Como resultado deste trabalho, obteve-se uma métrica de validação que utiliza uma aproximação do método de correlação cruzada para lidar com o componente da fase do erro e uma aproximação ao método de Deformação Dinâmica do Tempo (DTW) para lidar com componentes da magnitude e topologia. Bons resultados foram alcançados quando esta métrica foi comparada com validações baseadas em sistemas inteligentes.

Mongiardini *et al.* (2013) proporcionam uma visão geral sobre os passos que devem ser seguidos para realizar a verificação e validação dos modelos matemáticos usados na simulação de acidentes em rodovias. Para a execução deste processo utilizam-se como métricas principais o *Sprague & Geers* e ANOVA. Este artigo dá algumas dicas de como trabalhar com critérios de aceitação para estas métricas, baseando-se na análise da dispersão típica dos resultados de provas similares.

Dvurecenska *et al.* (2018) propuseram outra métrica de validação que mescla o uso de umbrais baseados nas incertezas dos dados devido às medições com erro relativo normalizado, e que é robusta na presença de dados com ampla variação. O resultado desta métrica é a probabilidade que as predições do modelo representem o sistema real

com determinado nível de confiabilidade. Como vantagens, esta métrica permite trabalhar com dados de amplas variações, gera uma resposta mais informativa e quantitativa que procedimentos de validação anteriores, embora sejam qualitativamente equivalentes.

Outras referências mais antigas podem ser citadas, como (Russell, 1997b) que desenvolveu e aplicou a métrica Russell. Em (Ray, 1996) foi explicada a métrica ANOVA e em Yamaoka, Nakagawa e Uno (1978) foi descrito o critério de informação de Akaike (AIC). E o já citado trabalho de Morales Alvarado (2017), que desenvolveu um método de validação com um sistema *fuzzy* para avaliar os resultados obtidos.

2.2 Literatura referente à inteligência artificial

A área de Inteligência Artificial (IA) exibiu um crescimento exponencial nos últimos anos, o que motivou um grande número de pesquisas nessa área (BOUTABA *et al.*, 2018). Esta seção apresenta a literatura sobre os métodos utilizados nesta dissertação.

Collobert e Williamson (2001) propuseram a decomposição do algoritmo SVM-Torch, similar ao SVM-Light. Com este algoritmo solucionou-se de maneira eficiente problemas de regressão de grande escala. Além disto, foi comparado o desempenho do algoritmo proposto Nodelib, um outro algoritmo para solucionar problemas de regressão de grande escala, obtendo-se resultados que demonstram a convergência deste novo algoritmo.

Almuallim, Kaneda e Akiba (2002) descreveram detalhadamente a base teórica das Árvores de Decisão (DT), para então debater algumas de suas limitações e modos de evadi-las. Além disso, são descritos diferentes exemplos de aplicação destes métodos e ferramentas para aprimorar o desempenho dos algoritmos de Aprendizado de Máquina (ML).

(Wang, 2005) é um livro com 20 capítulos, escolhidos a partir da miríade de novas aplicações de SVM, poderosos algoritmos SVM e análises teóricas esclarecedoras. Escrito por especialistas em seus respectivos campos, os primeiros 12 capítulos concentram-se na teoria de SVM, enquanto que os 8 capítulos finais enfatizam aplicações práticas.

Yuchun Tang *et al.* (2009) inseriram alterações no algoritmo SVM para solucionar o problema das classes desequilibradas. Para tal, foram adicionados compensadores heurísticos distintos na modelagem SVM, incluindo custo de aprendizagem sensível, sub e sobre amostragem. Assim, estas estratégias são comparadas com outros métodos com

diferentes bases de dados usando várias métricas. Os resultados deste trabalho indicam um desempenho superior em termos de velocidade e precisão do método proposto.

Para o estudo da base teórica que explica os métodos vistos neste trabalho, foram usados três livros. No primeiro (RUSSELL; NORVIG; EDWARDS, 2010) é descrita uma introdução ao tópico da IA e são explicados conceitos e algumas ideias gerais sobre métodos. Com o intuito de aprofundar o conhecimento sobre algoritmos de aprendizagem, ter um maior detalhe da teoria e para ver exemplos de aplicações práticas, os livros de Haykin (2009) e Alpaydin (2010) também foram usados.

Franco-Arcega, Flores-Flores e Gabbasov (2013) utilizam DT para classificar objetos astronômicos. Neste caso, a ML é usada devido ao fato dos astrônomos trabalharem com grandes quantidades de dados, além do fato do método permitir o melhor entendimento do comportamento dos diferentes parâmetros em observação. O algoritmo usado ParDTLT demonstrou eficiência e alta concordância com os valores esperados.

Adebiyi, Adewumi e Ayo (2014) examinaram o desempenho de previsão do ARIMA e o modelo de redes neurais, com dados publicados da Bolsa de Valores de Nova York. Os resultados obtidos revelaram a superioridade do modelo de redes neurais (ANN) sobre o modelo ARIMA, ademais, também resolveram e esclareceram as posições contrárias relatadas na literatura sobre a superioridade das ANN sobre o modelo ARIMA e vice-versa.

Song e Lu (2015) listaram os algoritmos mais usados para gerar árvores de decisão incluindo Árvores de Classificação e Regressão (CART), C4.5 - Detecção Automática de Interação com chi-quadrado (CHAID) e Árvore Rápida, Imparcial, Eficiente e Estatística (QUEST); além disso, foram citados exemplos para visualizar as estruturas das árvores geradas usando os programas IBM SPSS Modeler® e SAS Enterprise Miner®.

Yuan *et al.* (2015) propuseram um método de detecção de falhas para atuadores de *ailerons* submetidos a condições variáveis. Neste trabalho, três ANN foram usadas para detecção e localização preliminar de falhas, para isto, foi preciso comparar o erro residual (valor da diferença entre a saída real e a estimada) e o limiar ou ainda, comparando a corrente do motor de força com sua corrente estimada. Considerando as condições variáveis, cargas aerodinâmicas foram introduzidas na rede neural e os espectros de ordem de treinamento foram projetados. Por fim, a eficácia do esquema proposto foi validada por um modelo de simulação com diferentes falhas.

Em (Liu *et al.*, 2015) quatro algoritmos híbridos para a decomposição da velocidade do vento foram propostos. Neste caso, o algoritmo Adaboost foi adotado para fornecer uma estrutura de treinamento híbrida e as redes neurais *perceptron* multicamadas foram construídas para realizar o cálculo de previsão. Vários métodos foram utilizados para treinar estas ANN e os resultados obtidos demonstraram que todos os algoritmos híbridos sugeridos foram apropriados para as previsões de velocidade do vento.

Behera *et al.* (2017) explicaram os conceitos e evolução da IA. Para tal, alguns dos algoritmos mais usados foram comparados, baseando-se em suas noções básicas. Usando a base de dados *Sentiment140*, foram treinados para realizar previsões de três métodos, seguido da comparação do desempenho destes através de parâmetros, como o tempo de treinamento, tempo de predição e acurácia. Os resultados deste artigo mostraram grande similitude nos parâmetros avaliados para os métodos DT, *Naive Bayes* e SVM.

Boutaba *et al.* (2018) e Stamp (2018) apresentam as vantagens da aplicação de IA em redes de computadores e compararam diferentes métodos. Os autores dão enfoque na área de tráfego de informações, otimização de desempenho e segurança da rede. Além disso, é apresentado um bom resumo sobre a história, explicações da base teórica e discussão sobre benefícios e limitações dos distintos algoritmos de ML usados na área de suporte técnico das redes.

2.3 Literatura referente ao controle preditivo baseado em modelos

Outro tema importante para este estudo é o MPC, abordado nesta seção, apresentando inicialmente as referências que oferecem uma visão do estado da arte deste método de controle. Além disto, referências a artigos e livros sobre teoria e aplicação deste método e trabalhos feitos na Planta Piloto de Neutralização de pH com MPC são citados.

Cutler e Ramaker (1980) propuseram um algoritmo de controle que incorpora controle por pré-alimentação (*feedforward*) e multivariável, com base nas informações provenientes de um modelo dinâmico da planta e que para minimizar a integral do erro em relação ao tempo, usa o método do erro quadrático mínimo; que foi chamado Controlador de Matriz Dinâmica (DMC). Com base no trabalho anterior, Garcia e Morshedi (1986) implantaram novas alterações no método e desenvolveram o Controlador de Matriz

Dinâmica Quadrática (QDMC), que tem como principal característica lidar com as restrições do sistema usando programação quadrática.

Porfírio, Almeida Neto & Odloak (2003) apresentaram o MPC como uma solução para controlar uma coluna de destilação com comportamento não linear. Para abordar esse comportamento foi estabelecido um modelo linearizado para cada um dos possíveis pontos de operação. Neste caso, são usados modelos baseados em degraus e o custo do controlador é minimizado para o pior modelo do processo. Os resultados do sistema são comparados com o sistema real trabalhando em condições típicas, com controladores funcionando nos modos servo e regulatório.

Entre os textos estudados sobre o MPC, pode-se citar Camacho e Bordons (2007), que explicam os elementos e algoritmos gerais do método, esquemas comerciais e aplicações mais específicas, tais como MPC multivariável, robusto, com restrições não lineares e híbrido. Bemporad, Ricker e Morari (2019b, 2019a) explicaram o trabalho com o *Toolbox* de controle preditivo do Matlab®, além de esclarecerem alguns aspectos teóricos básicos.

Favaro (2012) aplicou um controle preditivo aplicado na mesma planta piloto deste estudo, dividindo o seu trabalho em quatro etapas principais: implementação das malhas de controle regulatório, identificação dos modelos, construção do controlador preditivo, aplicações e análises experimentais, onde foram comparados os diferentes resultados da planta quando submetida a um controlador PID, quando aplicado um controlador preditivo em cascata com os controladores PID e quando se usa apenas o controlador preditivo com ação direta nos atuadores. Nota-se que para o pH, que é altamente não linear, o controlador MPC demonstrou um desempenho mais eficiente que o controlador PID.

Darby e Nikolaou (2012), Xi, Li e Lin (2014) e Forbes *et al.* (2015) descreveram o estado da arte do MPC nas áreas acadêmica e industrial. Os principais tópicos discutidos abordam os desafios para usar esta tecnologia devido às suas limitações, recomendações de aplicação, principais características e referências para o estudo desta ferramenta.

Morales Alvarado (2013) fez a identificação e modelagem da Planta Piloto de Neutralização de pH, visando desenvolver o projeto de controladores nessa planta. A estimativa dos modelos foi feita de dois modos: (1) estimar modelos fenomenológicos que representem o comportamento de todo o sistema, incluindo os controladores PID do processo e (2) identificar modelos do processo com os dados coletados dos sinais de controle e as variáveis de saída do processo. Com os modelos estimados do processo projetou-se uma estratégia de controle MPC do tipo QDMC, envolvendo dois esquemas de controle. O

primeiro esquema calculou os *setpoints* ótimos que ingressaram nas malas de controle do processo. O segundo esquema calculou os sinais de controle ótimos que ingressaram diretamente nos atuadores do processo. Como resultado desse trabalho, pôde-se aferir a superioridade do sistema de controle QDMC usando as duas configurações, com respeito ao uso só do controlador PID. O teste em modo regulatório permitiu observar o desempenho e a robustez do algoritmo QDMC, sendo este mais rápido e apresentando menor sobressinal em comparação com o controlador PID atuando sozinho.

Vozák & Veselý (2014) projetaram, modelaram e implementaram um MPC robusto. O método de projeto baseado em desigualdades matriciais bilineares e lineares (BMI e LMI) é reduzido a um cálculo de ganho de realimentação *off-line*. Para mostrar a eficiência do método, um exemplo numérico e uma aplicação real foram apresentadas.

Romanus & Giovaninni (2014) realizaram um estudo implementando e testando diferentes algoritmos de controle de pH aplicados na Planta de Neutralização de pH do LCPI. Entre os algoritmos estão os controladores PID, PID-2DOF, controlador por pré-alimentação (*feedforward*), controle adaptativo e MPC. No modo servo, o MPC apresentou resultados significativamente melhores em termos de tempo de acomodação e sobressinal máximo, quando comparado aos outros algoritmos. No entanto, este controlador não alcançou um desempenho satisfatório quando se aplicou um degrau negativo de 7 para 6 no valor de referência do pH, além de que no modo regulatório o MPC apresentou o pior índice de desempenho da integral do erro quadrático ponderado pelo tempo (*ITSE*).

Em Downey *et al.* (2017), pela necessidade de se ter um controle mais robusto da Qualidade Crítica dos atributos, foi criado um sistema altamente automatizado, que periódica e eficientemente gera modelos de predição, usando diferentes técnicas de Identificação de Sistemas. Estes modelos foram usados pelos MPC que controlam processos de cultivos celulares da indústria biofarmacêutica. Como resultado desse trabalho melhorou-se o desempenho dos MPC nestes processos e possibilitou-se o seu uso em outros processos farmacêuticos.

Lorenzen, Cannon e Allgöwer (2019) desenvolveram um controlador robusto com restrições aplicado a um sistema linear com incertezas paramétricas e perturbações aditivas. Esse trabalho introduz uma estrutura matematicamente rigorosa e computacionalmente tratável para estabilizar o controle preditivo do modelo com a estimativa de parâmetros *on-line* para melhorar o desempenho e reduzir o conservadorismo. Um exemplo numérico comprova as propriedades e eficácia propostas nesse trabalho.

3 MÉTODOS DE VALIDAÇÃO

A validação é definida como o teste que certifica a validade e correção dos dados introduzidos, segundo as regras e padrões estabelecidos. Na área de Engenharia de Sistemas este conceito é usado nas tarefas de Identificação e de Controle, comumente intituladas Validação de Modelos e Avaliação de Malhas de Controle e, neste trabalho são usados seus resultados para conformar um conjunto de dados de treinamento. Este capítulo apresenta os métodos de validação usados para solucionar ambas as tarefas.

3.1 Validação de modelos matemáticos

Os modelos computacionais e modelos matemáticos são ferramentas muito usadas na engenharia há mais de 60 anos. A sua criação precisa de diferentes procedimentos (Figura 3-1); dentre eles a Verificação e a Validação. A Verificação “é o processo para determinar se o modelo representa adequadamente a base matemática ou física do sistema modelado” (KAT; ELS, 2012; LIU *et al.*, 2011). A Validação “é o processo para determinar o grau de precisão com que o modelo representa adequadamente o sistema modelado” (LIU *et al.*, 2011; KAT; ELS, 2012). O conhecimento e implantação destes processos são fundamentais em Identificação de Sistemas, em que é estudado como modelar e analisar sistemas a partir de valores medidos (AGUIRRE, 2000; PACHECO DA SILVA, 2015).

A Figura 3-1 ilustra as propostas de (THACKER *et al.*, 2004) e (KAT; ELS, 2012), ela mostra um resumo das tarefas e etapas associadas ao processo de verificação e validação. Este esquema destaca a existência de duas etapas no processo de validação:

- 1) Comparar os dados obtidos nos experimentos e os dados obtidos na simulação usando uma técnica de validação; e
- 2) Verificar se o resultado da etapa anterior atende aos requisitos de precisão desejados.

Antes de comparar, é preciso escolher uma técnica de validação que gere as informações desejadas; por isso, essa escolha é essencial para concluir corretamente o primeiro passo da validação de modelos. Há vários métodos de validação, que são divididos em dois grupos fundamentais: quantitativos e qualitativos. Ambos podem ser usados para validar modelos; mas por terem características diferentes, são normalmente usados em situações distintas. Pode-se dizer que os métodos quantitativos e qualitativos se complementam, nenhum deles é perfeito e ambos têm vantagens e desvantagens.

Assim, o processo de seleção não é trivial e é preciso fundamentar as escolhas. Nesta dissertação os métodos quantitativos têm um papel fundamental. Esta seção descreve os métodos de validação usados.

Figura 3-1- Visão geral dos processos de verificação e validação de modelos.

Fonte: (THACKER *et al.*, 2004).

3.1.1 Métodos Quantitativos

Os métodos quantitativos estimam um valor que descreve o nível de concordância entre os dados preditos e os dados medidos. Estes métodos são objetivos, significando que o valor resultante não muda, independentemente do analista, o que ocorre com os métodos qualitativos. Esta é a principal característica destas métricas e o que as torna um complemento das métricas qualitativas (OBERKAMPF; BARONE, 2006; LIU *et al.*, 2011). As métricas quantitativas têm um baseamento matemático e são comumente divididas em três categorias (OBERKAMPF; BARONE, 2006):

- 1) Estimaco de parmetros:  focada em obter uma resposta numrica que descreve quo prximos esto os dados preditos e os dados medidos. Esta categoria inclui Mtricas Adimensionais da Mdia do Erro (MAME), Mtricas de Distncia (MD) e Mtricas de Limiar.
- 2) Testes de hiptese: so focadas em obter uma resposta Sim ou No, os dados preditos e medidos tm uma certa consistncia com um dado nvel de significncia. Para isso, so usados testes estatsticos no paramtricos e verifica-se que os dados possuem uma certa distribuio.
- 3) Anlise Bayesiana:  focada em avaliar a fidelidade da fsica do modelo computacional existente. Usa a anlise probabilstica como a principal ferramenta.

As categorias 1 e 2 so usadas nesta dissertao pelo fato de que elas no s ajudam na validao, mas tambm na deteco de possveis erros (verificao) e na melhoria dos modelos (OBERKAMPF; BARONE, 2006). Outro fator  que os resultados so numricos, o que  muito til em etapas subsequentes  validao, como  mostrado nos captulos seguintes. Esses tipos de mtricas quantitativas provaram ser bem-sucedidas em trabalhos como (VAN WOENSEL; VANDAELE, 2006; PILLONETTO; CHIUSO; NICOLAO, 2011; LUMLEY; SCOTT, 2015; e MORALES ALVARADO, 2017).

3.1.1.1 Teste de hipteses

Nos testes de hipteses, a hiptese nula H_0  comparada com a hiptese alternativa H_1 para saber qual das duas descreve melhor o conjunto de dados. Como resultado deste tipo de teste, se pode: rejeitar a hiptese H_0 em favor da hiptese H_1 ou no rejeitar a hiptese H_0 . No rejeitar H_0 no significa que ela seja verdadeira, apenas que no h evidncias suficientes para rejeit-la. Uma das desvantagens desse tipo de teste  que nunca se pode provar que a hiptese  verdadeira, apenas que  menos eficiente para descrever os dados do que outras hipteses (SHESKIN, 2000; SORNETTE *et al.*, 2007).

Entre os testes de hipteses, esto os testes de significncia, nos quais uma nica hiptese nula H_0  considerada e a hiptese alternativa H_1 so todas as outras hipteses possveis. Neste caso, a resposta fornecida pelo teste  que os dados examinados no contradizem a hiptese H_0 . Outra maneira de interpretar esta resposta  que a hiptese H_0 descreve os dados com um certo nvel de confiana (tambm chamado de nvel de significncia) (SORNETTE *et al.*, 2007).

Devido s caractersticas mencionadas nesses testes, o tratamento dos erros  feito pela classificao em dois tipos de erros (MORALES ALVARADO, 2017):

- Tipo I: ocorre quando a hipótese H_0 é rejeitada quando é realmente verdadeira; e
- Tipo II: ocorre quando a hipótese H_0 é aceita quando realmente não deve ser aceita.

O nível de significância α é a probabilidade de aceitar a hipótese H_0 como verdadeira quando na realidade ela é falsa (probabilidade de um erro do Tipo I). Este nível de significância é arbitrariamente escolhido pelo usuário. Os valores mais comuns são 5%, 1% e 0,1% (DEGROOT; SCHERVISH, 2012; MORALES ALVARADO, 2017). Nesta dissertação irá se trabalhar com um nível de significância de 5%.

Para rejeitar e aceitar as hipóteses, um método muito comum é o cálculo do valor p (p -value) usando os dados observados. Tecnicamente, esse valor representa a probabilidade de observar um resultado como rejeitável. Neste trabalho, um p -value maior ou igual a 0,05 (5%) significa que a hipótese H_0 não deve ser rejeitada. Se o valor de p -value for menor, deve ser rejeitado (MORALES ALVARADO, 2017).

3.1.1.1.1 TESTE DE KOLMOGOROV-SMIRNOV (K-S)

O teste de Kolmogorov–Smirnov (K-S) é projetado para ser usado com variáveis contínuas. Este teste é catalogado como um teste de dados ordinais, visto que se precisa construir uma função de distribuição cumulativa (CDF) (SHESKIN, 2000). O teste supõe que dada uma população que tem uma CDF hipotética $F_0(x)$, em que para cada um dos valores de x o valor de $F_0(x)$ é a razão entre os valores da população que são menores ou iguais a x , a CDF empírica de uma amostra aleatória de N elementos (observações) deve ficar bastante perto da CDF ($F_0(x)$) hipotética. Se não fica suficientemente perto, isto é uma evidência que a distribuição hipotética não é a correta (MASSEY, 1951).

Sendo assim, se $F_0(x)$ é a CDF hipotética da população e $S_n(x)$ é a CDF observada da amostra (i.e. $S_n(x) = k/N$, onde k é número de elementos da amostra menores ou iguais a x), então pode-se calcular D (Equação (3.1)), independente de $F_0(x)$, se $F_0(x)$ é contínua (MASSEY, 1951; MATHWORKS, 2006b).

$$D = \max |F_0(x) - S_n(x)| \quad (3.1)$$

Há tabelas com os valores críticos de D segundo o nível de significância (α) para amostras de distintos tamanhos (N). Neste trabalho se usa um nível de significância de 95% ($\alpha=0,05$) e amostras com valores de N tendendo ao infinito. Assim, pode-se definir $D_{crítica} = 1.36/\sqrt{N}$ (MASSEY, 1951). Para comprovar se a função de distribuição

hipotética assumida e a distribuição empírica são próximas, é feito o seguinte teste de hipótese:

- H_0 : Os dados observados seguem uma função de distribuição contínua $F_0(x)$.
- H_1 : Os dados observados não seguem uma função de distribuição contínua $F_0(x)$.

A hipótese nula, H_0 , é rejeitada com um nível de significância se o valor estatístico (D) obtido for maior que o valor crítico ($D_{crítico}$) obtido (MORALES ALVARADO, 2017).

3.1.1.1.2 TESTE DE ANDERSON-DARLING (A-D)

O teste de Anderson–Darling (A-D) é normalmente usado para comprovar a distribuição normal. Mas pode ser usado com outros tipos de distribuições hipotéticas, embora não se tenham todos os parâmetros da distribuição. Este teste estatístico pertence à família de funções estatísticas de distribuições empíricas quadráticas, as quais medem as distâncias (d) entre a distribuição hipotética $F(x)$ e a CDF empírica $F_n(x)$ como:

$$d = n \int_{-\infty}^{\infty} (F_n(x) - F(x))^2 w(x) dF(x) \quad (3.2)$$

$$w(x) = [F(x)(1 - F(x))]^{-1} \quad (3.3)$$

onde os valores são ordenados $x_1 < x_2 < \dots < x_n$, $w(x)$ é uma função custo e n é a quantidade de elementos da amostra. A estatística de Anderson-Darling é calculada usando a Equação (3.4) (MATHWORKS, 2006c; ANDERSON, 2010).

$$AD = -N - \sum_{i=1}^N \frac{2i-1}{N} [\ln(F(X_i)) + \ln(1 - F(X_{N+1-i}))] \quad (3.4)$$

Segundo a Equação (3.4), $F(x)$ é o tipo de distribuição assumida e x_i é o valor da amostra. Dada a estatística de A-D, formula-se a hipótese nula a rejeitar:

- H_0 : os dados observados seguem uma função de distribuição específica $F(x)$.
- H_1 : os dados observados não seguem uma função de distribuição específica $F(x)$.

A hipótese nula é rejeitada se o valor da estatística de A-D (AD) for maior que o valor crítico ($AD_{crítico}$), formulado pela Equação (3.5); caso contrário a hipótese é aceita (MORALES ALVARADO, 2017).

$$AD > AD_{crítico} = 0.01265 + \frac{0.1757}{n} \quad (3.5)$$

3.1.1.1.3 TESTE DE SHAPIRO-WILK (S-W)

O teste de Shapiro-Wilk (S-W) é baseado em uma correlação das amostras estatísticas ordenadas $x_1, x_2, \dots, x_n$, onde se assume que elas seguem uma distribuição normal. Simulações de Monte Carlo têm provado que o poder de análise deste teste é o melhor para uma significância dada, seguido de perto pelo A-D (comparação feita entre os testes S-W, K-S, Lilliefors e A-D) (RAZALI; WAH, 2011). A estatística S-W (W) é definida pela Equação (3.6).

$$W = \frac{\sum_{i=1}^n a_i x_i^2}{\sum_{i=1}^n x_i - \bar{x}} \quad (3.6)$$

sendo $\bar{x}$ a média dos dados observados e a_i são constantes geradas pela média, pela variância e pela covariância dos dados amostrados. Os valores de a_i são definidos como as componentes do seguinte vetor ("The Shapiro-Wilk and related tests for normality", 2015):

$$\alpha = M^* V^{-1} \left((M^* V^{-1}) (V^{-1} M) \right)^{-1/2} \quad (3.7)$$

onde M corresponde aos valores ordenados esperados da distribuição normal padrão para uma amostra de dimensão N e V corresponde à matriz de covariância. Definida a estatística W , as hipóteses nulas e alternativa são formuladas a seguir:

- H_0 : os dados observados seguem uma função de distribuição normal.
- H_1 : os dados observados não seguem uma função de distribuição normal.

A hipótese nula é rejeitada se o valor de W calculado for menor que $W_{critico}$, podendo este valor ser consultado na Tabela A.5 de (SHESKIN, 2000), que contém valores de $W_{critico}$ até $n = 50$ (número de elementos da amostra). Para lidar com amostras maiores, pode-se consultar (BEN SAIDA, 2014)(BEN SAIDA, 2014)(ROYSTON, 1992; RAHMAN; GOVINDARAJULU, 1997; BEN SAIDA, 2014). O valor de W é limitado entre 0 e 1 e espera-se que, para aceitar fortemente a hipótese nula, a estatística W convirja a 1. Caso se obtenha um valor de W significativa-mente pequeno, é aceita a hipótese alternativa (MORALES ALVARADO, 2017).

3.1.1.1.4 TESTE DE WILCOXON (W_x)

O teste de Wilcoxon (*Wilcoxon Signed Rank Test*) é um teste não paramétrico para a mediana da diferença, que precisa que a diferença seja aproximadamente simétrica e que os dados medidos estejam em intervalos constantes. Este teste é mais poderoso, detectando mais diferenças entre duas populações que o t-teste. Para calcular o teste de Wilcoxon para cada item de uma amostra de n elementos, calcula-se a diferença entre

dois pares de elementos, Dx_i . O resultado desta subtração precisa ser processado: usa-se somente o valor absoluto ($|Dx_i|$) e omitem-se os valores de diferenças iguais a zero, com estes valores cria-se uma nova amostra com uma população n' ($n' \leq n$). Atribuir *rankings* R_i com valores desde 1 até n' para cada um dos valores $|Dx_i|$, atribuindo o *ranking* $R_i = 1$ para o menor valor de $|Dx_i|$ e n' para o maior valor de $|Dx_i|$. Se dois ou mais valores $|Dx_i|$ são iguais, atribuir a cada um deles a média dos *rankings* que eles teriam individualmente se não tivesse ocorrido o empate; no final colocar os sinais para cada *ranking* atribuído, R_i , dependendo do sinal original de Dx_i . Para calcular a estatística do teste de Wilcoxon (Wx) usa-se a Equação (3.8) (BERENSON; LEVINE; KREHBIEL, 2012).

$$Wx = \sum_{i=1}^{n'} R_i^{(+)} \quad (3.8)$$

Para calcular os valores críticos, onde $n' \leq n$, pode-se procurar a Tabela A.5 de (SHESKIN, 2000) ou a Tabela 12.19 de (BERENSON; LEVINE; KREHBIEL, 2012). Para obter os valores críticos para $n' > 20$, a estatística Wx aproxima-se da distribuição normal com média μ_W (Equação (3.9) e desvio padrão σ_W (Equação (3.10) (SHIER, 2004; BERENSON; LEVINE; KREHBIEL, 2012; MORALES ALVARADO, 2017).

$$\mu_{Wx} = \frac{n'(n' + 1)}{4} \quad (3.9)$$

$$\sigma_{Wx} = \sqrt{\frac{n'(n' + 1)(2n' + 1)}{24}} \quad (3.10)$$

Para realizar o teste de Wilcoxon, calcula-se uma nova estatística (Z) mostrada na Equação (3.11).

$$Z_{stat} = \frac{Wx - \mu_{Wx}}{\sigma_{Wx}} \quad (3.11)$$

Definida a estatística de Wilcoxon, a hipótese nula e alternativa são formuladas a seguir:

- H_0 : o valor absoluto da diferença entre duas amostras é simétrico em torno de zero.
- H_1 : o valor absoluto da diferença entre duas amostras não é simétrico em torno de zero.

Finalmente, para avaliar as hipóteses propostas, a estatística de Wilcoxon Z é comparada com o valor crítico, a um nível de significância α , utilizando a Tabela A.1 de

(SHESKIN, 2000). Se a estatística de Wilcoxon for maior que o valor crítico de Z_α , a hipótese nula é rejeitada (MORALES ALVARADO, 2017).

3.1.1.2 Métricas Adimensionais da Média do Erro

As Métricas Adimensionais da Média do Erro (MAME) são métricas de validação quantitativas, baseadas em definir a correspondência entre alguma medida estatística de tendência central do erro e uma outra medida estatística dos dados medidos e/ou preditos (WILLMOTT; ROBESON; MATSUURA, 2012). O resultado deste tipo de métrica pode fornecer uma indicação absoluta ou relativa do relacionamento entre os dois conjuntos de dados. Para fazer essas comparações, três características fundamentais podem ser avaliadas: amplitude, fase e topologia. Este item exhibe este tipo de métricas de validação.

3.1.1.2.1 EFICIÊNCIA DO MODELO

A eficiência do modelo (EF), proposta em (LOAGUE; GREEN, 1991), é um índice que avalia o resultado da simulação de um modelo, comparando dois vetores de dados, permitindo medir a discrepância entre os valores medidos e estimados pelo modelo. Usa a avaliação da soma quadrática de suas diferenças, o quadrado da média do erro (MSE) para comparar a concordância entre dois conjuntos de dados, segundo a Equação (3.12):

$$MSE = \frac{1}{N} \sum_{t=1}^N [y(t) - \hat{y}(t)]^2 \quad (3.12)$$

sendo $y(t)$ e $\hat{y}(t)$ os valores medidos e estimados de dimensão N cada um. O coeficiente de performance (CP) é obtido ao se dividir o valor de MSE pela variação individual do processo; este é aproximadamente zero quando os dois conjuntos de dados apresentam concordância. Com base no uso do CP é gerado o índice de Eficiência (EF), que é definido pela Equação (3.13).

$$EF = 1 - \sqrt{CP} = 1 - \sqrt{\frac{\sum_{i=1}^n [y(t) - \hat{y}(t)]^2}{\sum_{i=1}^n [y(t) - \bar{y}(t)]^2}} \quad (3.13)$$

Para tornar mais robusto o índice EF , extrai-se a raiz quadrada de CP , para penalizar o comprimento das diferenças dos vetores medidos e estimados. O resultado final do índice EF indica que valores iguais a 1 são considerados uma aproximação perfeita dos valores medidos e estimados. À medida que o índice EF fica menor, a inconsistência entre os dados reais e estimados é maior (MORALES ALVARADO, 2017).

¹ A eficiência do modelo EF é igual ao índice de ajuste (FIT) proposto em (LJUNG; 1999).

Valores maiores do que 0,6 descrevem modelos aceitáveis na maioria dos trabalhos que usam esta métrica em aplicações de Identificação de Sistemas.

3.1.1.2.2 MÉTRICA DE THEIL

O Coeficiente de Desigualdade de Theil (*TIC*) (vide Equação (3.14)) se baseia na medida intitulada raiz quadrada média do erro relativo (RMSPE), a qual considera as proporções de desigualdade que os erros representam devido ao *bias*, variância e covariância (THEIL, 1966; VAN WOENSEL; VANDAELE, 2006; MORALES ALVARADO, 2017).

$$TIC = \frac{\sqrt{\frac{1}{N} \sum_{t=1}^N (y(t) - \hat{y}(t))^2}}{\sqrt{\frac{1}{N} \sum_{t=1}^N (y(t))^2 + \frac{1}{N} \sum_{t=1}^N (\hat{y}(t))^2}} \quad (3.14)$$

O valor do índice *TIC* varia entre 0 (para previsões perfeitas) e 1 (para previsões totalmente imperfeitas). Esta métrica pode ser encarada como proposto em (ROWLAND; HOLMES, 1978): para valores inferiores a 0,4 o modelo é aceitável; para valores entre 0,4 e 0,7 o modelo é moderadamente aceitável e para valores maiores que 0,7 o modelo é rejeitado.

Como já foi mencionado, o índice *TIC* possui limite superior e inferior, isto é valorizado como uma vantagem para trabalhar com erros muito grandes, de acordo com (THEIL, 1966; VAN WOENSEL; VANDAELE, 2006). Ainda assim, o índice *TIC* apresenta duas limitações vistas em (BLIEMEL; MACKAY, 1973):

- 1) O fato de não ser calculado apenas pela raiz quadrada do erro médio faz com que o índice seja fortemente influenciado pelas características dos dados preditos.
- 2) O índice *TIC* não fornece informações significativas quando se trabalha com modelos altamente incongruentes.

3.1.1.2.3 MÉTRICA DE WILMOTT

O índice de concordância proposto em (WILLMOTT, 1982), denominado por *d*, é uma medida descritiva de tipo relativo e delimitado, que calcula a taxa entre o *MSE* e o erro potencial (*PE*) (MORALES ALVARADO, 2017). Deste modo, o índice *d* determina as diferenças que atingem as magnitudes e sinais dos valores estimados com relação a $y(t)$ e aos desvios estimados em relação a $\bar{y}(t)$. Assume-se que $y(t)$ e $\bar{y}(t)$ não apresentem erro algum (WILLMOTT, 1982).

$$d = 1 - \frac{\sum_{t=1}^N (y(t) - \hat{y}(t))^2}{\sum_{t=1}^N (|\hat{y}(t) - \bar{y}(t)| + |y(t) - \bar{y}(t)|)^2} \quad (3.15)$$

A Equação (3.15), proposta em (WILLMOTT, 1982), descreve o índice d , o qual foi usado com sucesso em muitas aplicações, mas na medida em que estas aplicações foram diversificando, também foi diversificando o seu desempenho. Notou-se que o fato de elevar ao quadrado a soma dos erros influência de forma não balanceada o resultado final (WILLMOTT; ROBESON; MATSUURA, 2012). Tentado solucionar essas desvantagens e obter um resultado mais fácil de interpretar, foi proposto em (WILLMOTT; ROBESON; MATSUURA, 2012) uma nova interpretação deste índice (vide a Equação (3.16)). Neste caso, os limites desta métrica são -1 e 1. Pode ser interpretada do seguinte modo: um resultado de 0,5 significa que a média do erro é a metade do desvio padrão dos dados medidos, mas se o valor for -0,5 então a média do erro é o dobro do desvio padrão. Segundo esta interpretação pode ser definido que os modelos com um resultado maior do que 0,6 são considerados aceitáveis.

$$d' = \begin{cases} 1 - \frac{\sum_{t=1}^N |y(t) - \hat{y}(t)|}{2 \sum_{t=1}^N |y - \bar{y}(t)|}, & \text{se } \sum_{t=1}^N |y(t) - \hat{y}(t)| < 2 \sum_{t=1}^N |y - \bar{y}(t)| \\ \frac{2 \sum_{t=1}^N |y - \bar{y}(t)|}{\sum_{t=1}^N |y(t) - \hat{y}(t)|} - 1, & \text{se } \sum_{t=1}^N |y(t) - \hat{y}(t)| > 2 \sum_{t=1}^N |y - \bar{y}(t)| \end{cases} \quad (3.16)$$

3.1.1.2.4 MÉTRICA DE RUSSELL

A métrica de Russell (RUSSELL, 1997) compara as respostas de dois sistemas usando a análises da fase, magnitude dos vetores e com os resultados destas análises determina um erro global. Para analisar a magnitude usa-se a Equação (3.17), chamada de erro relativo de magnitude (RME) (RUSSELL, 1997).

$$RME = \frac{\sum_{i=1}^N \hat{y}_i(t)^2 - \sum_{i=1}^N y_i(t)^2}{\sqrt{\sum_{i=1}^N \hat{y}_i(t)^2 \cdot \sum_{i=1}^N y_i(t)^2}} \quad (3.17)$$

A análise do erro relativo de fase P_r da métrica de Russell é baseada no produto ponto a ponto entra as duas fases dos vetores normalizados. A Equação (3.18) é usada para obter P_r , que fica limitado entre 0 e 1; sendo que P_r igual a 0 corresponde a uma correlação direta entre os vetores $\hat{y}(t)$ e $y(t)$; e um valor de P_r igual a 1 equivale a dizer que se têm fases diferentes (uma defasagem de $-\pi$) (RUSSELL, 1997).

$$P_r = \frac{1}{\pi} \cos^{-1} \left(\frac{\sum_{i=1}^N \hat{y}_i(t) \cdot y_i(t)}{\sqrt{\sum_{i=1}^N \hat{y}_i(t)^2 \cdot \sum_{i=1}^N y_i(t)^2}} \right) \quad (3.18)$$

O cálculo de erro global (C_r) é feito usando os dos tipos de erro relativos previamente citados (Equação (3.20)), mas no caso do rme são feitas modificações (vide a Equação (3.19)) para que o valor fique com uma escala aproximada com P_r (RUSSELL, 1997).

$$M_r = \text{sign}(rme) \cdot \log_{10}(1 + |rme|) \quad (3.19)$$

$$C_r = \sqrt{\frac{\pi}{4}(M_r^2 + P_r^2)} \quad (3.20)$$

Na Equação (3.20) foi usada a constante de normalização $\pi/4$ para não se afetar o erro de magnitude, de fase ou global um com outro, mantendo sem *bias* a métrica. Com este valor de escala sugere-se 0,2 como um valor aceitável para comparações gerais. Esta constante de normalização pode ser alterada dependendo dos objetivos da avaliação do sistema (RUSSELL, 1997).

3.1.1.2.5 MÉTRICA DE SPRAGUE E GEERS

A métrica de Sprague e Geers para comparar respostas de diferentes sistemas foi apresentada em (GEERS, 1984). Este trabalho propõe uma métrica que atribui um valor numérico que determina as discrepâncias entre as respostas, considerando a magnitude e a fase das respostas. Uma outra característica importante desta proposta é o fato que as discrepâncias de magnitude e fase não se afetam mutuamente.

A Equação (3.21) apresenta como é calculado o fator da magnitude do erro, $M_{S\&G}$. Os valores, neste caso, têm limites entre -1 e infinito; sendo 0 o valor obtido quando se tem duas respostas idênticas, mas entendendo que mesmo que as respostas não sejam idênticas, o índice $M_{S\&G}$ pode ser igual a 0.

$$M_{S\&G} = \sqrt{\frac{\sum_{i=1}^N \hat{y}_i(t)^2}{\sum_{i=1}^N y_i(t)^2}} - 1 \quad (3.21)$$

A Equação (3.22) mostra como é calculado o fator da fase do erro, $P_{S\&G}$. Esta equação não foi proposta em (GEERS, 1984), o cálculo do fator de erro da fase foi alterado pelo fato que a proposta original não tinha a sensibilidade que se precisava. Portanto, usa-se esta nova expressão proposta em (SPRAGUE; GEERS, 2004).

$$P_{S\&G} = \frac{1}{\pi} \cos^{-1} \left(\frac{\sum_{i=1}^N \hat{y}_i(t) \cdot y_i(t)}{\sqrt{\sum_{i=1}^N \hat{y}_i(t)^2 \cdot \sum_{i=1}^N y_i(t)^2}} \right) \quad (3.22)$$

O erro global $C_{S\&G}$ é calculado pela Equação (3.23)

$$C_{S\&G} = \sqrt{(M_{S\&G}^2 + P_{S\&G}^2)} \quad (3.23)$$

Esta métrica é muito similar à de Russell, mas com uma abordagem diferente para quantificar o erro associado à magnitude. A principal diferença é que os resultados terão maior dispersão e são mais influenciados pelo erro de magnitude. Olhando o princípio desta métrica e da métrica de Russell, pode ser escolhido o valor de 0,4 como o valor limite. Isto significa que a razão das amplitudes dos sinais sempre será menor que a metade e a defasagem menor do que $\pi/2$.

3.1.1.2.6 MÉTRICA ANOVA

Esta métrica foi proposta em (RAY, 1996). Para a sua aplicação assume-se que das respostas que simulam o mesmo evento, suas diferenças são atribuídas ao erro aleatório dos experimentos. Se este fato é correto, então os resíduos são aleatórios, assim a distribuição destes deveria ter uma distribuição normal com média zero (RAY, 1996). Baseado nesta ideia, é usada a Equação (3.24) para calcular a estatística-t de um teste bicaudal.

$$T_A = \frac{\bar{e}}{\sigma_c / \sqrt{n}} \quad (3.24)$$

A variável $\bar{e}$ é a média dos resíduos das respostas, σ_c é o desvio padrão do resíduo e n é o número de amostras. Com os resultados de T_A pode-se chegar à seguinte conclusão: se $-t_{\alpha, n-1} \leq T_A \leq t_{\alpha, n-1}$, onde $t_{\alpha, n-1}$ são os valores críticos da estatística-t para n amostras que podem ser encontrados em (SHESKIN, 2000), então não há uma diferença significativa entre as duas respostas a um nível de confiança α ; portanto não é possível rejeitar a hipótese nula. Esta prova estatística não comprova que as duas respostas sejam idênticas (RAY, 1996).

Em (MONGIARDINI *et al.*, 2013) propõe-se as Equações (3.25) e (3.26) para calcular $\bar{e}$ e σ_c , respetivamente. Neste caso, o resíduo é dividido pelo valor máximo do conjunto de dados m_{max} , obtendo-se uma normalização aproximada dos resíduos.

$$\bar{e} = \frac{\sum_{i=1}^n ((y_i(t) - \hat{y}_i(t)) / m_{max})}{n} \quad (3.25)$$

$$\sigma_c = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (e_i - \bar{e})^2} \quad (3.26)$$

Finalmente, o teste de ANOVA conduz à avaliação dos seguintes critérios (RAY, 1996):

- A média dos resíduos entre dois conjuntos de dados deve ser essencialmente zero;

- O desvio padrão dos resíduos deve ser menor que algum valor razoavelmente imposto; e
- O valor do p -value obtido da estatística T_A deve ser maior que o nível de significância da distribuição-t para duas caudas e um número de amostras n .

3.1.1.3 Métrica de Dvurecenska

A métrica de validação de Dvurecenska mescla o uso de limiares baseados nas incertezas de medição e simulação com uso de erro relativos normalizados, além disto é robusta na presença de grandes variações dos dados (DVURECENSKA *et al.*, 2018). Para fazer isto, trabalha-se com dois tipos de incerteza: incerteza de decomposição u_{deco} e incerteza de medição u_{cal} . A u_{deco} é declarada na Equação (3.27), onde n é a quantidade de dados e $r(t)$ é o *setpoint* e a u_{cal} é obtida mediante os distintos procedimentos de calibração dos instrumentos.

$$u_{deco}^2 = \frac{1}{n} \sum_{t=1}^n (r(t) - y(t))^2 \quad (3.27)$$

A equação usada nesta dissertação difere um pouco da original porque esta métrica foi projetada para uso com imagens de corpos sólidos mecânicos. No caso da equação original, o índice MSE é calculado entre a imagem decomposta e a imagem real, enquanto a equação modificada calcula o MSE entre o *setpoint* $r(t)$ e a saída do processo $y(t)$, devido ao trabalho em uma malha fechada. A incerteza total u_{exp} é calculada com a Equação (3.28) (DVURECENSKA *et al.*, 2018).

$$u_{exp} = \sqrt{u_{deco}^2 + u_{cal}^2} \quad (3.28)$$

A métrica de Dvurecenska pode ser dividida em quatro passos:

- 1) Calcular o erro relativo normalizado $e_k(t)$ como se observa na Equação (3.29), onde $\max|y(t)|$ é o maior valor medido.

$$e_k(t) = \left| \frac{\hat{y}(t) - y(t)}{\max|y(t)|} \right| \quad (3.29)$$

- 2) Calcular o peso de cada erro $w_e(t)$ (Equação (3.30)), onde n é a quantidade de dados.

$$w_e(t) = \frac{e_k(t)}{\sum_{t=1}^n e_k(t)} * 100 \quad (3.30)$$

- 3) Definir o limiar do erro e_{th} pela Equação (3.31).

$$e_{th} = \frac{2u_{exp}}{\max|y(t)|} * 100 \quad (3.31)$$

- 4) Calcular a métrica de Dvurecenska (VM), como a soma dos pesos do erro menores que o umbral do erro. A Equação (3.32) apresenta este procedimento, onde o operador indica que se soma só quando o valor de $w_e(t)$ é menor que e_{th} .

$$VM = \sum_t w_e(t) ||_{e_k(t) < e_{th}} \quad (3.32)$$

Este somatório representa a probabilidade do erro normalizado ser igual ou menor que a incerteza do experimento; dito em outras palavras, o índice VM representa a probabilidade do modelo ser uma boa representação da realidade para um caso dado. Esta métrica não depende apenas dos erros entre os valores medidos e previstos, mas também é influenciada pelas incertezas do processo: quanto maiores as incertezas, pior será a confiabilidade da métrica. Para aplicações gerais, modelos com $VM > 60\%$ são considerados aceitáveis.

3.1.1.4 Métrica da Deformação Dinâmica do Tempo (DTW)

O método da Deformação Dinâmica do Tempo (DTW) mede a semelhança de dois conjuntos de dados (também chamados de sequência) e pode ser catalogada como uma Métrica de Distância. Foi proposto pela primeira vez nos anos 1960 como uma ferramenta para reconhecer padrões de áudio e, com o tempo, tornou-se popular e usado em diferentes áreas, como: mineração de dados, medicina e visão computacional. Diferentemente dos demais métodos apresentados, ele não calcula ponto a ponto a distância entre os valores dos conjuntos, mas calcula usando técnicas de comparação “um contra todos” e “todos contra um”; outra característica sua é a de encontrar o “melhor caminho” entre as duas sequências. Os artigos (SAKOE; CHIBA, 1978) e (SALVADOR; CHAN, 2004) foram usados como diretrizes neste trabalho.

A primeira etapa na implementação deste algoritmo é construir uma matriz de custos acumulados (Ψ) de dimensão $N \times M$, em que N e M são as dimensões dos conjuntos de dados X e Y , respectivamente. A distância entre os pontos x_i e y_j é armazenada em cada uma de suas células. A partir de Ψ , o “melhor caminho” (Figura 3-2) é definido como a sequência $W = \{w_1, w_2, \dots, w_k\}$ em que $\max(N, M) \leq K \leq (N + M - 1)$ e $w_k = \Psi[i_k, j_k]$, para que as seguintes condições sejam atendidas (SARIN *et al.*, 2009):

- Condições de contorno: $w_1 = \Psi[1, 1]$ e $w_k = \Psi[N, M]$, W começa em um canto e deve cruzar toda a matriz para o canto diagonal oposto, certificando-se de que não resta nenhum elemento para ser avaliado.

- Continuidade: se $w_{k-1} = \Psi[i_{k-1}, j_{k-1}]$ e $w_k = \Psi[i_k, j_k]$, então $i_k - i_{k-1} \leq 1$ e $j_k - j_{k-1} \leq 1$, confirmando que cada passo do caminho é adjacente ao anterior.
- Monotonia: se $w_{k-1} = \Psi[i_{k-1}, j_{k-1}]$ e $w_k = \Psi[i_k, j_k]$, então $i_k - i_{k-1} \geq 0$ e $j_k - j_{k-1} \geq 0$, isso força o caminho a estar sempre à frente.

Figura 3-2- Exemplo de matriz de custos acumulados.

Fonte: (SALVADOR; CHAN, 2004).

A construção da matriz é feita através de uma programação dinâmica do tipo $O(N^2)$, que preenche as células usando os critérios mostrados na Equação (3.33),

$$\Psi[i, j] = dw[i, j] + \min(\Psi[i - 1, j - 1], \Psi[i - 1, j], \Psi[i, j - 1]) \quad (3.33)$$

sendo dw a distância entre os pontos da sequência (função custo), normalmente calculada usando a distância euclidiana, como é o caso do trabalho de (SARIN *et al.*, 2009), embora possa ser usado outro tipo de distância, como o tipo de *Manhattan*, usada neste trabalho. A Equação (3.34) descreve como calcular as duas distâncias, basta levar em conta que para a distância euclidiana $p = 2$ e para a distância de *Manhattan* $p = 1$.

$$dw[i, j] = \left(\sum_{i,j=1}^{N,M} |x_i - y_j|^p \right)^{\frac{1}{p}} \quad (3.34)$$

O fato de poder analisar a semelhança entre dois conjuntos de dados serve como motivação para usar este método usando a derivada das respostas temporais dos sistemas. Há vários métodos que abordam essa ideia, um deles é o proposto em (GÓRECKI; ŁUCZAK, 2013). Na Equação (3.35) é descrita a função custo usada neste método, onde p é escolhido dependendo do tipo de distância a ser calculada, α é o fator de ajuste do algoritmo entre $[0, \pi/2]$, x'_i e y'_j são as derivadas discretas de x_i e y_j .

$$dw[i, j] = (1 - \alpha) \left(\sum_{i,j=1}^{N,M} |x_i - y_j|^p \right)^{\frac{1}{2}} + \alpha \left(\sum_{i,j=1}^{N,M} |x'_i - y'_j|^p \right)^{\frac{1}{2}} \quad (3.35)$$

A distância entre os conjuntos de dados e o melhor caminho de acordo com o método *DTW* são calculadas usando as Equações (3.36) e (3.37), respectivamente.

$$DTW(x, y) = \Psi[N, M] \quad (3.36)$$

$$DTW_{opt}(x, y) = \min_W \left\{ \sum_{k=1}^K w_k \right\} \quad (3.37)$$

A *DTW* não é afetada pelas diferenças de fase ou de frequência dos dados e pode ser interpretada como quanto menor o valor, maior é a semelhança dos conjuntos de dados. Como desvantagem, pode-se mencionar que este método não está de acordo com o triângulo da desigualdade; portanto, não pode ser considerada como uma métrica (SARIN *et al.*, 2009). Ademais, no caso de se trabalhar com a derivada da sequência, esta é muito sensível ao ruído, por isso recomenda-se a filtragem dos dados muito ruidosos (GÓRECKI; ŁUCZAK, 2013). Pelo fato que *DTW* é considerada uma forma de medição da distância entre duas sequências, neste trabalho se define que uma distância aceitável é de 2% da amplitude do sinal de excitação utilizado para validar os modelos.

3.1.1.5 Critérios de seleção de modelos

Os critérios de seleção são métodos para escolher o modelo que melhor descreve um certo conjunto de dados. Estes critérios medem quanta informação perde o modelo quando tenta simular ou representar o conjunto de dados; aquele que perde menos informações é o melhor. Outra aspecto importante destes critérios é serem baseados no princípio da parcimônia, significando que o melhor modelo é o menos complexo, que consegue representar o conjunto de dados (BOZDOGAN, 1987; FABOZZI *et al.*, 2014).

Alguns dos critérios mais comuns de seleção de modelos são: *AIC* (*Akaike Information Criterion*) (AKAIKE, 1973, 1998); *FPE* (*Final Prediction Error*) (AKAIKE, 1969); *BIC* (*Bayesian Information Criterion*) (SCHWARZ, 1978). “Estes critérios de informação comparam o ganho de se ter mais informação em termos da redução da variância nas inovações, com o custo de se obter esse benefício em termos da perda de graus de liberdade na estimação” (MORALES ALVARADO, 2017). Estes, com exceção do *FPE*, tomam a forma de uma função de verossimilhança penalizada e um termo de penalização, que está em função do número de parâmetros do modelo. Conforme foi dito, a seleção se baseia no menor valor que produz o critério de informação (MORALES ALVARADO, 2017).

As Equações (3.38), (3.39) e (3.40) descrevem os critérios *FPE*, *AIC*, *BIC*, respectivamente. Para os critérios *AIC* e *BIC*, os termos de penalização são $2n_p$ e $n_p \cdot$

$\ln(N)$, respectivamente; o resto da equação corresponde à função de verossimilhança. No caso do *FPE* a ideia é a mesma, mas se lida com a função custo do sistema para avaliar o desempenho do modelo.

$$AIC = N \cdot \ln \left(\det \left(\frac{1}{N} \sum_{t=1}^N \varepsilon(t, \hat{\theta}_N) \cdot (\varepsilon(t, \hat{\theta}_N))' \right) \right) + N (n_y (\ln(2\pi) + 1)) + 2n_p \quad (3.38)$$

$$FPE = \det \left(\frac{1}{N} \sum_{t=1}^N \varepsilon(t, \hat{\theta}_N) \cdot (\varepsilon(t, \hat{\theta}_N))' \right) \left(\frac{1 + n_p/N}{1 - n_p/N} \right) \quad (3.39)$$

$$BIC = N \cdot \ln \left(\det \left(\frac{1}{N} \sum_{t=1}^N \varepsilon(t, \hat{\theta}_N) \cdot (\varepsilon(t, \hat{\theta}_N))' \right) \right) + N (n_y (\ln(2\pi) + 1)) + n_p \cdot \ln(N) \quad (3.40)$$

O parâmetro N é o número de dados de identificação; $\varepsilon(t, \hat{\theta}_N)$ é o vetor de dimensão $n_y \times 1$ que contém os erros de predição; n_p é o número de parâmetros estimados; n_y é o número de saídas do modelo (MATHWORKS, 2019a, 2019b).

3.1.1.6 Validação da dinâmica do modelo

A validação dinâmica de modelos consiste em avaliar a resposta do modelo ante diferentes estímulos no domínio do tempo ou da frequência. Estes tipos de validação são bem comuns na validação de modelos paramétricos, pois fornecem informações básicas sobre a dinâmica do modelo. Os estímulos mais usados são o degrau e o impulso no domínio do tempo e a senoide no domínio da frequência.

No domínio do tempo é comum excitar o modelo com uma entrada em degrau. A resposta em malha aberta permite verificar se o modelo estabiliza ou não, calcular o ganho do sistema e estimar as principais características da resposta transitória²: tempo de subida (t_r), tempo de acomodação (t_s) e sobressinal máximo (M_p). Para a malha fechada pode-se calcular estes mesmos elementos, além do erro estacionário (e_{ss}).

3.2 Avaliação das malhas de controle

Avaliar o bom funcionamento das malhas de controle é uma maneira de validá-las. Esta etapa é essencial nos processos de fabricação; pois, apesar desses sistemas serem projetados para operar automaticamente, podem ser afetados por falhas. O comportamento dos sistemas de controle é um indicador claro do desempenho que uma determinada planta pode ter. É por isso que é enfatizada nas diferentes indústrias o uso

² A definição destes conceitos pode ser encontrada em (Ogata, 2010).

de índices para monitorar estes comportamentos. Trabalhos como (RUEL, 2002), (GRUPPE, 2012) e (MORALES ALVARADO; GARCIA, 2018) são exemplos de como estas questões são tratadas, tanto na área produtiva quanto na acadêmica.

Segundo o objetivo principal desta dissertação, é preciso ter uma série de índices que avaliem o resultado das técnicas de controle que serão usadas como o conjunto de saídas do algoritmo de IA. Com esse objetivo em mente e seguindo os postulados vistos em (GRUPPE, 2012), propõe-se uma série de índices para descrever, de modo simples, o comportamento do processo e, assim, avaliar o desempenho da malha de controle.

3.2.1 Métricas estatísticas

Os métodos de estatísticas descritivas são comumente usados para interpretar, analisar, organizar e descrever conjuntos de dados; funcionalidades usadas nos setores industrial e acadêmico. A capacidade destas métricas para descrever a tendência central e a dispersão dos dados é a principal causa de seu emprego neste trabalho. Algumas das mais conhecidas são a média, variância, desvio padrão e variabilidade.

A variabilidade (*var*) é uma das métricas estatísticas que descreve a razão entre o desvio padrão (σ) e a média (μ) de um sinal (LIPTÁK, 2006). É aplicada para examinar o comportamento da variável controlada. Ela dá uma ideia de quão disperso é o sinal e sobre a estabilidade do sistema. Quando este índice é zero, significa que o sistema não apresenta oscilações, à medida que se afasta do zero, a variável controlada oscila com amplitudes cada vez maiores. A Equação (3.41) mostra como calcular a variabilidade.

$$var = \frac{2\sigma}{\mu} * 100\% \quad (3.41)$$

Considerando que se trabalhe em malha fechada, tanto na fase de identificação quanto na fase de controle, e que o interesse neste tipo de situação é conhecer a dispersão dos dados em relação ao *setpoint*, algumas destas métricas populares podem ser reinterpretadas para uma melhor descrição da situação estudada. A Equação (3.42) descreve a média do erro relativo da saída do processo com respeito à referência (*MESR*) e a Equação (3.43) é a média da razão entre a diferença dos valores consecutivos da saída e a referência (*MRSR*), que quantifica a tendência central da variação da saída do processo a cada instante em relação ao instante anterior, sendo $y(t)$ e $r(t)$ os valores da saída do processo e o *setpoint* nos instantes t e $t - 1$, $\mu(r(t), r(t - 1))$ é a média dos valores de *setpoint* nos instantes t e $t - 1$.

$$MESR = \frac{1}{N} \sum_{t=1}^N \left(\left| \frac{y(t) - r(t)}{r(t)} \right| \right) \quad (3.42)$$

$$MRSR = \frac{1}{N} \sum_{t=1}^N \left(\left| \frac{y(t) - y(t-1)}{\mu(r(t), r(t-1))} \right| \right) \quad (3.43)$$

3.2.2 Índice de Harris

O índice de Harris (HARRIS, 1989) pode ser usado para medir o desempenho das malhas de controle estacionárias. Este índice representa a razão entre a variância mínima teórica (σ^2_{MV}) e a variância da saída do processo (σ^2_y). Uma das limitações deste método é que σ^2_{MV} não é fácil de se obter ou geralmente não está disponível. Por esse motivo, em (SHUNTA, 1995) foi proposto um método baseado na fórmula de *Fellner* para calculá-lo. As Equações (3.44) a (3.46) mostram como calcular o índice de Harris (η_{Harris}).

$$\sigma_{MV'} = \sigma_{cap} * \sqrt{2 - \left(\frac{\sigma_{cap}}{\sigma_{tot}} \right)^2} \quad (3.44)$$

$$\sigma_{cap} = \sqrt{\frac{\sum_{t=2}^n y(t) - y(t-1)}{2(n-1)}} \quad (3.45)$$

$$\eta_{Harris} = \frac{\sigma_{MV'} + s}{\sigma_{tot} + s} * 100 \quad (3.46)$$

O parâmetro $\sigma_{MV'}$ é o desvio mínimo teórico, σ_{tot} é o desvio padrão total da saída, σ_{cap} é a capacidade do desvio padrão, n é a quantidade de elementos do vetor $y(t)$, $y(t)$ representa os dados de saída do processo no instante t , s é o fator de sensibilidade, considerado como 0,1% do valor da escala da variável (MORALES ALVARADO, 2017).

Quando este índice é 100%, significa que o processo atinge a variância mínima teórica. Para o uso deste índice, recomenda-se calculá-lo primeiro com o sistema operando em condições ideais. A partir deste valor, compare-o com os valores obtidos durante a operação normal da planta. Este índice é útil para determinar como o comportamento da saída muda devido a distúrbios aleatórios e não fornece informações específicas sobre alterações no *setpoint*, distúrbios conhecidos, tempo de acomodação, decaimento de oscilações ou estabilidade (JOE QIN, 2002; RICE; JYRINGI; COOPER, 2006).

3.2.3 Índice Watchdog

Este índice foi proposto em (RHINEHART, 1995). É usado para monitorar o desvio entre a variável do processo e o *setpoint*, seu aumento indica que o desvio é maior entre as duas variáveis. As Equações (3.47) a (3.49) descrevem esse índice, que se baseia no cálculo da razão entre a variação da diferença entre o *setpoint* e a variável de processo e a variação dos valores consecutivos da variável de processo; sendo que $y(t)$ representa os dados de saída do processo no instante t , $r(t)$ representa o *setpoint* no instante t , N é a quantidade de elementos do vetor das variáveis, r_{wi} é o índice *Watchdog*.

$$S_1^2 = \frac{1}{(N-1)} \sum_{t=1}^N (y(t) - r(t))^2 \quad (3.47)$$

$$S_2^2 = \frac{1}{2(N-1)} \sum_{t=1}^N (y(t) - y(t-1))^2 \quad (3.48)$$

$$r_{wi} = \frac{S_1^2}{S_2^2} \quad (3.49)$$

Este índice, assim como o índice de Harris, não fornece uma ideia absoluta, apenas uma ideia comparativa. Portanto, o mesmo valor deste índice pode ter significados diferentes para diferentes processos. Essa métrica também é influenciada pelo comportamento da variável do processo, mas não fornece nenhuma informação específica sobre a estabilidade do sistema (REYES DREKE; PÉREZ SERRANO; GARCIA, 2020).

3.2.4 Integral Absoluta da Ação de Controle

A integral absoluta da ação de controle (*IAU*) é uma métrica que segue a lógica das chamadas métricas da integral de erro. Neste caso, o sinal da saída do controlador (u) é integrado. Como os índices anteriores, este é comparativo e não absoluto; e isso ajuda a quantificar e comparar o esforço de controle (u) realizado pelos sistemas. A Equação (3.50) descreve essa métrica; sendo $u(t)$ o esforço de controle no instante t e N a quantidade de elementos do vetor das variáveis.

$$IAU = \sum_{t=1}^N |u(t)| \quad (3.50)$$

4 TRABALHO NA PLANTA PILOTO DE NEUTRALIZAÇÃO DE pH

A Planta Piloto de Neutralização de pH, do LCPI, é uma instalação para pesquisa acadêmica, na qual são realizados trabalhos voltados para a área de controle de processos, de modelagem fenomenológica e de identificação de sistemas. Ela tem três variáveis principais a serem controladas: pH, nível e temperatura.

Este capítulo descreve a operação da planta, suas principais características e equipamentos; incluindo uma descrição dos aspectos fundamentais da teoria de seus métodos de controle. Além disso, é descrito o projeto de identificação realizado na planta.

4.1 Descrição da Planta Piloto de Neutralização de pH do LCPI

Na planta há dois tanques que armazenam soluções ácida e básica, intitulados Tanque de Ácido Primário (TAP) e Tanque de Ácido Secundário (TAS) (vide Figura 4.1), cujos níveis são monitorados por uma chave de nível alto e a entrada de água é liberada por válvulas solenoides. É importante ter uma boa mistura entre a solução ácida ou básica e a água, o que é feito por borbulhamento. A vazão de saída desses tanques entra no Tanque de Ácido Primário Intermediário (TAPI) e no Tanque de Ácido Secundário Intermediário (TASI), respectivamente. Nesses tanques, o nível é regulado usando boias instaladas na entrada. A principal função do TAPI e do TASI é fornecer um fluxo constante para o Tanque do Reator (TR), que é regulado por válvulas manuais e medido por medidores de vazão magnéticos. Para neutralizar o pH, a solução ácida vinda do TAPI e do TASI é misturada no TR com uma solução básica preparada no Tanque de Base da Bomba (TBB), que fornece solução básica para o TR via uma bomba dosadora. No tanque TBB, o nível é controlado por meio de chaves de nível alto e baixo.

No TR, onde a mistura das soluções ácida e básica ocorre por meio de um agitador, o pH, o nível e a temperatura são medidos e controlados. O nível é regulado por uma válvula pneumática com posicionador eletropneumático, que ajusta a vazão de saída e é medido por um medidor de pressão diferencial. A temperatura é medida com um Pt-100 e é controlada por dois elementos: o primeiro, responsável pelo aquecimento é uma resistência elétrica de 5 kW e o segundo, responsável pelo resfriamento é uma válvula de injeção

de água fria. O pH é medido por um pHmetro e é controlado pelo ajuste da vazão de base vinda do TBB. A planta é controlada e monitorada por um SDCD da ABB, que é equipado com um sistema supervisor IIT800xA, que permite comunicação de alto desempenho via protocolo MMS, permitindo a troca de dados entre eles (CORREA DE GODOY, 2012).

Figura 4-1- Diagrama P&ID simplificado da Planta Piloto de Neutralização de pH.

Fonte:(GARCIA MARQUES, 2015).

A planta possui um sistema de segurança que evita situações de emergência, que é executado por um CLP da Allen-Bradley (SLC-500). Maiores detalhes sobre a operação da planta são encontrados em (CORREA DE GODOY, 2012; PÉREZ SERRANO, 2019).

4.1.1 Malhas de controle

Na planta, além das três malhas de controle principais, existem malhas de pressão, vazão e condutividade. Uma lista das variáveis medidas e manipuladas é mostrada nas Tabela 4.1 e 4.2. Dentre todas essas malhas, este trabalho focará no pH e no nível.

Tabela 4.1- Lista das variáveis manipuladas.

Variáveis Manipuladas	Tag
Válvula de Saída do Tanque Reator	(LV-16C)
Válvula de Entrada de Ácido Principal no Tanque Reator	(FV-31)
Válvula de Entrada de Ácido Secundário no Tanque Reator	(FV-32)
Válvula de Entrada de Água no Tanque Reator	(FV-33)
Válvula de Controle de Vazão de Base	(FV-40)
Válvula de Pressurização do TBV	(PV-71)
Válvula de Despressurização do TBV	(PV-72)
Bomba Dosadora	(FZ-41)
Resistência de Aquecimento	(TZ-21)
Agitador mecânico do TR	(MZ-81)

Fonte: AUTOR.

Tabela 4.2- Lista das variáveis medidas.

Variáveis de entrada	Tag
pH do TR	(AITY-40)
Condutividade do TR	(CITY-51)
Temperatura do TR	(TT-20)
Nível do Tanque Reator	(LIT-10)
Nível do Tanque de Base para a Válvula	(LIT-14)
Pressão no tanque TBV	(PIT-70)
Vazão de Ácido Principal	(FIT-31)
Vazão de Ácido Secundário	(FIT-32)
Vazão de Base pela Bomba	(FZ-41)

Fonte: AUTOR.

4.1.1.1 Controle de pH no Tanque Reator TR

A malha de controle de pH é composta por um transmissor indicador de pH modelo PH200 da Yokogawa (AITY-40), calibrado na faixa de 0 a 14 e com um erro de calibração de $\pm 0,15$ pH (YOKOGAWA, 1997). O atuador usado é uma bomba dosadora, que fornece um fluxo de solução alcalina que vem do TBB. A bomba é acionada por um sinal de 4-20 mA e opera por pulsos, tendo um comportamento linear. Seu fluxo instantâneo pode ser estimado pela Equação (4.1). Em experimentos feitos em (PÉREZ SERRANO, 2019), foi mostrado que os valores mínimos e máximos de pH que são alcançados são 2 e 12³.

$$Q = \frac{\text{stroke}(\%)}{100} \cdot \frac{\text{sin}(\%)}{100} (\text{Vazão Nominal Máxima}(lph)) \quad (4.1)$$

4.1.1.2 Controle de nível no Tanque Reator TR

O nível é medido por um transmissor indicador de nível modelo EJA-110A da Yokogawa (LIT-10), calibrado de 0 a 100% da altura do tanque e com um erro de calibração de $\pm 0,01$ % (YOKOGAWA, 2012). Este envia um sinal de 4-20 mA ao controlador no SDCD, o qual gera um sinal de saída de 4-20 mA enviado a um posicionador eletropneumático, que transforma o sinal de 4-20 mA em porcentagem de 0 a 100% de abertura da válvula.

O nível máximo e mínimo é limitado por problemas físicos e de segurança. O valor máximo é de 85%, pois o tanque possui um dreno nessa altura e, quando esse nível é atingido, o controlador encarregado da segurança bloqueia as entradas de água no TR. O valor mínimo é de 15%, nesse valor a medição de pH perde confiabilidade, pois é a altura em que o sensor de pH está instalado. Não se recomenda usar o agitador próximo ao valor do nível mínimo, pois isso atrapalha a medição do nível. Caso a resistência de aqueci-

³ O fato de atingir esses valores máximos e mínimos também é influenciado pelos tipos de ácido e base utilizados e suas concentrações (PÉREZ SERRANO, 2019).

mento do TR esteja conectada, a altura do fluido deve ser superior a 45%, caso contrário, o controlador encarregado da segurança (TZC-21) desconecta a resistência. Essas limitações definem a faixa de operação do nível entre 45-80% (CORREA DE GODOY, 2012).

4.1.2 Especificações da Planta Piloto de Neutralização de pH

A Tabela 4.3 resume as especificações de controle da planta piloto estabelecidas por (PÉREZ SERRANO, 2019) e descreve a faixa de operação das variáveis da planta. Este trabalho é focado na região próxima aos *setpoints*.

Tabela 4.3- Especificações da Planta Piloto de Neutralização de pH.

	Variáveis de processo			Variáveis de perturbação	
	pH	Nível	Temperatura	Vazão	Rotação
Variáveis controladas	AITY-40	LIT-10	TT-20	-	-
Variáveis manipuladas	FZ-41	LV-16C	TY-21 FV-33	-	-
Variáveis de perturbação	-	-	-	FIT-31 e FIT-32 MZ-81	
Controladores	AC-40	LC-16	TC-20	-	-
Limites de operação	2-12	45-80 [%]	25-40 [°C]	40-60 [l/h]	400-1000 [rpm]
Regiões de operação	2.45-3; 4.8-7.2; 7.2-9.4; 11.48-11.94	50-75 [%]	25-40 [°C]	-	-
Setpoint	7	65%	30°C	-	-
Ponto de operação	2.73; 7; 8.3; 11.71	65%	30°C	50 l/h	800 rpm
Objetivo de controle	Regulatório	Regulatório	Regulatório	-	-
Prioridade das variáveis	1	2	3	-	-

Fonte: (PÉREZ SERRANO, 2019).

4.1.3 Métodos de controle

Esta planta é equipada com uma rede que permite a comunicação do SDCD da ABB com os computadores de controle e supervisão da planta (Figura 4-2). O servidor de conectividade (CSFF01) desta rede tem instalado um serviço *OPC (OLE for Process Control)*, que permite a comunicação direta dos computadores com os equipamentos da planta usando o SDCD como ponte. Com estas ferramentas, pode-se não apenas controlar a planta usando os controladores PID implementados no SDCD, mas também controlá-la usando um controlador MPC em Matlab®, a partir dos computadores EW01 e OW02.

Figura 4-2- Diagrama da arquitetura da rede ethernet no LCPI.

Fonte: (DESVAGES; STABILITO, 2018).

4.1.3.1 Controladores PID

Os controladores PID são uma solução eficiente (em muitos casos) e altamente popular no setor industrial. Esta técnica provou ser capaz de lidar com um grande número de processos, apesar de sua relativa simplicidade. Há vários tipos de PID e, portanto, várias formas matemáticas de declará-lo. Em (OGATA, 2010) e (GARCIA, 2017) pode-se encontrar várias representações destes controladores, incluindo os clássicos. É muito comum que seus fabricantes usem fórmulas ligeiramente diferentes das normalmente citadas em ambientes acadêmicos. Neste caso, a ABB propõe a função de transferência vista na Equação (4.2) (ABB, 2012).

$$PID(s) = \left(G_c * \frac{MV_{max} - MV_{min}}{PV_{max} - PV_{min}} \right) \left(\beta * SP - PV + \frac{1}{s * TI} + \frac{s * TD}{1 + s * TF} \right) \quad (4.2)$$

O parâmetro G_c é o ganho proporcional, MV_{max} e MV_{min} são os valores máximo e mínimo da variável manipulada (saída do controlador), PV_{max} e PV_{min} são os valores máximo e mínimo da variável de processo (saída do processo), β é a constante do *set-point*, SP é o *set-point*, TI é o tempo integral, TD é o tempo derivativo, TF é a constante de tempo do filtro.

Os parâmetros de sintonia dos controladores vistos na Tabela 4.4 são fruto do trabalho desenvolvido em (PÉREZ SERRANO, 2019) e (PÉREZ SERRANO *et al.*, 2019), onde a sintonia dos controladores foi feita usando vários métodos, comparando-os e escolhendo-se o melhor. A configuração final do controlador foi um PI sintonizado pelo método de Ziegler e Nichols⁴, portanto os valores de TD e TF são 0 e 1, respectivamente.

Tabela 4.4- Parâmetros de sintonia dos controladores PID da planta.

Malhas de controle	Parâmetros do controlador PI				
	β	G_c	TI	TD	TF
pH	1	24.8085	166.66	0	1
Nível	1	19.7136	100	0	1

Fonte: (PÉREZ SERRANO, 2019).

4.1.3.2 Controlador MPC

O termo MPC não se limita a uma única estratégia de controle; ao contrário, descreve um grupo que compartilha a seguinte metodologia (CAMACHO; BORDONS, 2007):

1. Usando um modelo do processo, predições são feitas para cada instante de tempo, N passos à frente. As saídas futuras preditas ($y(t + k | t)$) dependerão dos valores do processo anteriores ao instante atual e de uma ação de controle futura ($u(t + k | t)$).
2. Para calcular a ação de controle, uma função custo é otimizada, cujo objetivo é manter a saída do processo a mais próxima possível da trajetória da referência fornecida ($r(t + k)$). A solução para este problema de otimização é explícita se for um modelo linear sem restrições, caso contrário, para achar a solução são usados métodos iterativos.
3. Somente o primeiro passo da ação de controle predita ($u(t + 1 | t)$) é enviado, enquanto isso é enviado, o próximo sinal de controle é calculado usando a nova saída real do processo, retornando à etapa 1.

⁴ São dois métodos de sintonia para controladores PID propostos em (ZIEGLER; NICHOLS, 1942). Na malha de pH usou-se o Método das Oscilações Contínuas e na malha de nível usou-se o Método de Curva de Reação.

Há vários modos de abordar essa metodologia, as usadas nesta dissertação são o MPC do tipo Controlador de Matriz Dinâmica (DMC) e sua variante Controlador de Matriz Dinâmica Quadrática (QMDC) para sistemas com restrições. Estes métodos propostos por (CUTLER; RAMAKER, 1979) e (GARCIA; MORSHEDI, 1986), respectivamente, foram projetados para lidar com modelos discretos de resposta ao degrau (FSR) e distúrbios constantes durante o horizonte de predição. A Equação (4.3) apresenta como as predições do processo são obtidas com esse tipo de modelo (CAMACHO; BORDONS, 2007).

$$\hat{y}(t+k|t) = \sum_{i=1}^k g_i \Delta u(t+k-i) + f(t+k) \quad (4.3)$$

O parâmetro k é o horizonte de predição⁵, Δu é o incremento da entrada do processo que neste caso coincide com a ação de controle, g_i são os coeficientes da resposta ao degrau, $f(t+k)$ é a resposta livre do sistema (parte da resposta que não depende da ação de controle futura) descrita pela Equação (4.4).

$$f(t+k) = y_m(t) + \sum_{i=1}^{\infty} (g_{k+i} - g_i) \Delta u(t-i) + \hat{d}(t+k|t) \quad (4.4)$$

onde $y_m(t)$ é o valor medido no instante de tempo t , $\hat{d}$ é o valor das perturbações preditas. Note que se o sistema for estável, os coeficientes g_i serão constantes a partir do período de amostragem N_x , então $g_{k+i} - g_i \approx 0, \forall i > N_x$. Se o sistema for instável, $N_x = \infty$ e, portanto, $f(t+k)$ não pode ser calculado (embora exista uma generalização, em caso de ter uma instabilidade produto de um integrador puro) (CAMACHO; BORDONS, 2007).

Para estimar as saídas do processo com um horizonte de predição p e um horizonte de controle m , utilizam-se as Equações (4.5) e (4.6), onde G é a matriz dinâmica do sistema, $\hat{y}$ é o vetor que contém as predições da saída, u é o vetor com os incrementos da ação de controle, f é o vetor que contém as respostas livres.

$$\hat{y} = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ g_2 & g_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_m & g_{m-1} & \dots & g_1 \\ \vdots & \vdots & \ddots & \vdots \\ g_p & g_{p-1} & \dots & g_{p-m+1} \end{bmatrix}_{p \times m} * \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+m-1) \end{bmatrix}_{m \times 1} + \begin{bmatrix} f(t+1) \\ f(t+2) \\ \vdots \\ f(t+p) \end{bmatrix}_{p \times 1} \quad (4.5)$$

$$\hat{y} = G * u + f \quad (4.6)$$

A função custo (J) deste algoritmo (Equação (4.7)) é baseada no erro quadrático. A partir disso, a ação de controle ($u(t)$) que a minimiza é calculada, ou seja, os valores para

⁵ Os horizontes de predição e de controle são a quantidade de períodos de amostragem que simulam o processo e a sequência de controle, respectivamente.

os quais a derivada é 0. Em caso de se estar perante um problema irrestrito e linear, então a solução analítica é mostrada na Equação (4.8).

$$J = \sum_{j=1}^p \gamma [\hat{y}(t+j|t) - r(t+j)]^2 + \sum_{j=1}^m \vartheta [\Delta u(t+j-1)]^2 \quad (4.7)$$

$$\mathbf{u} = (\mathbf{G}^T (\gamma \mathbf{I}_p) \mathbf{G} + \vartheta \mathbf{I}_m)^{-1} \mathbf{G}^T (\gamma \mathbf{I}_p) (\mathbf{r} - \mathbf{f}) \quad (4.8)$$

As constantes γ e ϑ são os pesos da saída do processo e da variável manipulada, respectivamente; r é a referência do sistema; $\hat{y}$ é a variável do processo predita pelo modelo; Δu é o incremento da variável manipulada; $\mathbf{u}$ é o vetor que contém os incrementos da ação de controle; p é o horizonte de predição; m é o horizonte de controle e $\mathbf{I}_p$ e $\mathbf{I}_m$ são as matrizes identidade de dimensão p e m , respectivamente.

Quando o problema tiver as restrições $\Delta u_{max} \leq u \leq \Delta u_{min}$, $u_{max} \leq u \leq u_{min}$, $y_{max} \leq y \leq y_{min}$, ele assume a forma de um problema de programação quadrática (QP). A otimização agora é numérica e, quando isto ocorre, o MPC é conhecido como QMDC. A Equação (4.9) mostra como estes problemas são formulados, onde $\mathbf{H}$ é a matriz hessiana vista na Equação (4.10), $\mathbf{\Gamma}$ é a matriz diagonal dos pesos da saída, $\mathbf{\Lambda}$ é a matriz diagonal dos pesos de entrada, $\boldsymbol{\delta}$ é o vetor do gradiente exibido na Equação (4.11) (BOUM; LATIFI; CORRIOU, 2013).

$$J = \min_{\Delta u(k)} \left[\frac{1}{2} \Delta u(t)^T \mathbf{H} \Delta u(t) - \boldsymbol{\delta}(t+1)^T \Delta u(t) \right] \quad (4.9)$$

$$\mathbf{H} = \mathbf{G}^T \mathbf{\Gamma}^T \mathbf{\Gamma} \mathbf{G} + \mathbf{\Lambda}^T \mathbf{\Lambda} \quad (4.10)$$

$$\boldsymbol{\delta}(t+1) = \mathbf{G}^T \mathbf{\Gamma}^T \mathbf{\Gamma} (\mathbf{r} - \mathbf{f}) \quad (4.11)$$

A Tabela 4.5 apresenta as restrições, parâmetros de configuração e sintonia do controlador MPC. Os resultados obtidos em (PÉREZ SERRANO, 2019) justificam seu uso.

Tabela 4.5- Parâmetros de configuração e sintonia do MPC.

Restrições e Parâmetros temporais	Símbolo	Malha	
		Nível	pH
Horizonte de predição	p	60	60
Horizonte de controle	m	2	2
Pesos da saída do processo	γ	0.3	0.3
Pesos da variável manipulada	ϑ	0.7	0.7
Valor máximo no valor atual da entrada	u_{max}	100	100
Valor mínimo no valor atual da entrada	u_{min}	0	0
Valor máximo no valor atual da saída	y_{max}	75	8.1
Valor mínimo no valor atual da saída	y_{min}	55	4.8
Limite superior na variação da entrada	Δu_{max}	100	100
Limite inferior na variação da entrada	Δu_{min}	-100	-100

Fonte: AUTOR.

4.2 Projeto de Identificação

A identificação de sistemas é um método em que, com base nos dados de entrada e saída de um processo, é estimado um modelo que descreve o sistema da melhor forma possível. Este tipo de procedimento tem sido amplamente estudado; duas das referências mais populares neste tópico são (SÖDERSTRÖM; STOICA, 1989) e (LJUNG, 1999). Eles propõem uma série de etapas para identificar os processos, mas atualmente, alguns softwares de identificação adicionam uma etapa de tratamento dos dados (Figura 4-3).

Figura 4-3- Fluxograma do processo de identificação.

Fonte: (LJUNG, 1999).

Segundo o proposto na Figura 4-3 foi criado um Projeto de Identificação⁶ na planta de pH usando o *software* CASI da startup Cursor Identificação e Controle. Neste trabalho, o controlador MPC atua nas entradas dos controladores PID existentes no SDCCD, motivo pelo qual os modelos descrevem o sistema em malha fechada e, assim, a identificação do sistema é feita nessa configuração. Os dados são coletados usando os *setpoints* da respectiva malha de controle como entrada e a saída é a variável do processo (nível e pH).

⁶ O projeto de Identificação neste trabalho é baseado em (PÉREZ SERRANO, 2019), no qual foi identificada a planta de pH, gerando bons resultados e facultando seu uso como modelos de referência da planta.

Para coletar os dados, dois experimentos foram executados usando sinais distintos de excitação, que se complementam: o primeiro usou um sinal degrau, que permite obter informações sobre as características do processo em regime transitório; o segundo usou o sinal GBN, projetado usando os resultados do primeiro experimento e é usado para calcular os parâmetros ótimos do modelo.

A Figura 4-4 mostra os resultados dos experimentos de identificação usando a excitação em degrau, trabalha-se na faixa de 60 a 70% para o nível e 6,5 a 7,5 para o pH.

Figura 4-4- Resultado dos experimentos de identificação com degrau na planta piloto.

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

O comportamento do nível em malha fechada é bastante similar a um sistema sub-amortecido de segunda ordem com atraso puro e com um pequeno sobressinal, elemento que é mais evidente quando o degrau negativo é aplicado. Uma das características da dinâmica deste processo é que ele tem um tempo de acomodação (t_s) de cerca de 125 s (critério de 2%) e um sobressinal de 1,2%. A resposta do pH se assemelha a um sistema superamortecido, embora na faixa de 7,5 ele tenha um comportamento diferente; mesmo assim, o tempo de acomodação é bastante semelhante, em torno de 100 s.

O GBN proposto por (TULLEKEN, 1990) é um sinal que alterna entre dois estados fixos em pontos discretos do tempo; tem propriedades estocásticas e segue uma distribuição de probabilidade de acordo com as Equações (4.12) e (4.13).

$$P[u(t) = -u(t-1)] = p_{sw} \quad (4.12)$$

$$P[u(t) = u(t-1)] = 1 - p_{sw} \quad (4.13)$$

O parâmetro p_{sw} é a probabilidade de comutação de nível. Para evitar variações muito rápidas, é definido um tempo mínimo de comutação T_{min} , que normalmente é escolhido entre 3 a 5 intervalos de amostragem T e um tempo médio de comutação $E(T_{sw})$ (switching time) dado pela Equação (4.14), em que quanto maior for o valor de p_{sw} maior a frequência de comutação do sinal GBN (MORALES ALVARADO, 2013).

$$E(T_{sw}) = \frac{T_{min}}{p_{sw}} \quad (4.14)$$

Usando os valores de t_s obtidos nos experimentos anteriores, o valor de T é recalculado e com isso é calculado o T_{min} do novo sinal de excitação. A Figura 4-5 apresenta o resultado dos experimentos de identificação para as malhas de controle de Nível e de pH.

Figura 4-5- Resultado dos experimentos de identificação com GBN na planta piloto.

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

Como parte da fase de “Tratamento de dados”, o *software* CASI da Cursor obtém a matriz de correlação em que se observa a relação entre as entradas e saídas do sistema, vista na Tabela 4.6. Os pares de entrada e saída indicados com “x” são correlacionados e para os quais deve ser identificado um modelo. Além disto, nesta fase o *software* remove *outliers*, filtra e normaliza os dados na faixa de -0,5 a 0,5 (vide as Figura 4-6 e 4-7).

Tabela 4.6- Matriz de correlação.

		Entradas		
		SP Nível	SP pH	Vazão de Ácido
Saídas	Nível	X	0	0
	pH	0	X	0

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

O processamento dos dados coletados eliminou um pouco de ruído, o que diminui as incertezas na hora de estimar e validar o modelo. De forma geral, as mudanças sofridas pelos dados nesta etapa foram pequenas, mantendo a integridade dos dados. Consequentemente, o processamento dos dados deveria melhorar a etapa de identificação.

Figura 4-6- Comparação entre os dados tratados e reais obtidos na malha de Nível.

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

Figura 4-7- Comparação entre os dados tratados e reais obtidos na malha de pH.

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

No processo de “Escolha da estrutura” e “Estimação de parâmetros” o *software* CASI lida com modelos polinomiais paramétricos, que podem ser descritos pela Equação (4.15).

$$y(t) = \frac{B(q, \theta)}{A(q, \theta)F(q, \theta)} u(t) + \frac{C(q, \theta)}{A(q, \theta)D(q, \theta)} e(t) \quad (4.15)$$

sendo A, B, C, D e F polinômios definidos pelas seguintes equações:

$$A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + \dots + a_{n_a} q^{-n_a} \quad (4.16)$$

$$B(q) = b_1 q^{-1} + b_2 q^{-2} + \dots + b_{n_b-1} q^{-n_b} \quad (4.17)$$

$$C(q) = 1 + c_1 q^{-1} + c_2 q^{-2} + \dots + c_{n_c} q^{-n_c} \quad (4.18)$$

$$D(q) = 1 + d_1 q^{-1} + d_2 q^{-2} + \dots + d_{n_d} q^{-n_d} \quad (4.19)$$

$$F(q) = 1 + f_1 q^{-1} + d_2 q^{-2} + \dots + f_{n_f} q^{-n_f} \quad (4.20)$$

onde n_a, n_b, n_c, n_d e n_f são os graus dos polinômios, também chamados de ordens do modelo. A Tabela 4.7 exibe um resumo das estruturas usadas pelo *software* CASI, cabendo destacar que ele faz a escolha da estrutura e ordem dos modelos de forma automática.

Tabela 4.7- Estruturas dos modelos não paramétricos.

Polinômio usado na Equação (4.15)	Nome da estrutura do modelo
A, B	ARX
A, B, C	ARMAX
B, F	Output Error (OE)
B, C, D, F	Box-Jenkins (BJ)

Fonte: (MORALES ALVARADO, 2013).

A validação dos modelos é feita automaticamente, o *software* utiliza uma série de técnicas para estimar o melhor modelo, entre elas a análise dos resíduos e as métricas do *FIT* e Theil. Usando um algoritmo sequencial e os valores das técnicas, um modelo do processo é escolhido. A Tabela 4.8 mostra os modelos obtidos e o resultado da validação.

Tabela 4.8- Resultado dos experimentos de identificação.

Malha de Controle	Modelo obtido	Tipo de estrutura	Intervalo de amostragem	Índice <i>FIT</i>	Índice Theil
Nível	$\frac{0,5002z^{-1} + 0,1185z^{-2}}{1 - 0,6886z^{-1} + 0,3072z^{-2}}$	OE	1 segundo	92.78%	0.055
pH	$\frac{0,8013z^{-1} - 0,5759z^{-2}}{1 - 0,8175z^{-1} + 0,04216z^{-2}}$	BJ	1 segundo	88.63%	0.059

Fonte: (CURSOR, 2019) e (PÉREZ SERRANO, 2019).

5 PROJETO DE INTELIGÊNCIA ARTIFICIAL

Como já mencionado, o objetivo deste trabalho é apresentar um algoritmo que classifique a qualidade dos modelos, com base nas distintas respostas de uma planta controlada por um MPC, quando se usam os modelos em questão. A qualidade destes sistemas de controle vai depender dos índices de desempenho (KPI) propostos no Capítulo 3. Quando se lida com Inteligência Artificial (IA) é recomendado comparar o desempenho de uma série de métodos e escolher aquele que atinja melhores resultados, porque até agora não há um método analítico para predizer de forma exata antes de treinar qual é o algoritmo que vai dar melhores resultados em uma certa situação. Neste trabalho são escolhidos três métodos: Árvores de Decisão (DT), Redes Neurais Artificiais (ANN), Máquina de Vetores de Suporte (SVM).

Na Figura 5-1 mostra-se o fluxograma do projeto de IA desta dissertação, que pode ser dividido em três etapas principais, com suas respectivas tarefas, as quais são:

- Etapa 1: foca no estudo da situação e do embasamento teórico que serão úteis para solucionar os problemas; a maior parte das tarefas desta etapa (“Implementar os KPI” e “Implementar os métodos de validação”) foi abordada no Capítulo 4, mas na Seção 5.1 expõe-se a base teórica dos algoritmos de IA escolhidos.
- Etapa 2: engloba duas tarefas fundamentais: obtenção e preparação do conjunto de dados para seu treinamento. Para obter os dados é preciso fazer vários experimentos, os quais são descritos na Seção 5.2. Seus resultados para cada uma das plantas são apresentados nos Apêndices A e B. Portanto, a Seção 5.2 está focada em explicar como foram interpretados os resultados dos experimentos e como foi preparado o conjunto de dados.
- Etapa 3: explicada nas Seções 5.4 e 5.4, apresenta os resultados do treinamento e da validação dos algoritmos obtidos de forma individual, que são comparados uns com os outros para conhecer qual gera uma melhor solução do problema.

5.1 Métodos de Aprendizagem Automática

A Inteligência Artificial é uma área de conhecimento amplamente explorada, remontando aos anos 1950 (CREVIER, 1993; MCCORDUCK, 2004).

Figura 5-1- Fluxograma do projeto de IA.

Fonte: AUTOR.

Desde a sua criação, a IA mostrou um grande potencial para resolver problemas distintos. Sua flexibilidade causou a sua divisão em vários ramos concentrados no estudo de diferentes algoritmos com propósitos similares. Um desses ramos é o Aprendizado de Máquina (ou *Machine Learning*, ML), que tem um alto nível de aceitação nos ambientes acadêmico e industrial. A redução do tempo de cálculo e o processamento de dados; a capacidade de lidar com tarefas de distintos níveis de complexidade; a grande quantidade de

dados que há hoje e o fato de ser relativamente simples de implementar⁷ são as principais causas da popularidade da ML (DAS; BEHERA, 2017; BOUTABA *et al.*, 2018).

A ML estuda algoritmos de programação capazes de otimizar seu desempenho usando dados de treinamento (também chamados de experimentos). Estes tipos de técnicas são geralmente usados quando existem situações em que se têm dados, mas não há algoritmos que descrevam a relação exata entre eles. Situações como essas são muito comuns na vida cotidiana, o que justifica a popularidade dessas técnicas. Diz-se que os algoritmos de ML têm a capacidade de aprender, pelo fato de se adaptarem ao conjunto de dados, não importa o quanto eles mudem. Estes algoritmos não se destinam a resolver uma situação específica, mas a encontrar um grupo de parâmetros que descrevam aproximadamente as relações entre os dados utilizados (ALPAYDIN, 2010).

Os métodos de ML podem ser classificados de várias formas. Dependendo do tipo de aprendizado, eles podem ser: Supervisionados, Não Supervisionados e Reforçado. Dependendo da função que executam, eles podem ser para: Classificação ou Regressão. Este trabalho é desenvolvido usando algoritmos de classificação com aprendizado supervisionado, visto que o conjunto de dados pode ser separado em entradas e saídas, e que esses dados de saída são valores discretos finitos, também conhecidos como classes.

5.1.1 Árvores de Decisão

As árvores de decisão (DT) são modelos hierárquicos de aprendizado supervisionado, nos quais as regiões locais correspondem a uma sequência de divisões recursivas em várias etapas finitas. Estes modelos são compostos de dois elementos principais: *nós de decisão internos* e *folhas*. Em cada nó de decisão m , é executada uma função $f_m(x)$ que discrimina os dados. As saídas da $f_m(x)$ são discretas e rotuladas, cada uma delas é intitulada *ramificação*. Basicamente, o algoritmo consiste em dado um conjunto de dados, é aplicado um teste em cada nó de decisão e, dependendo do resultado, os dados usam uma ou outra ramificação para ir ao próximo nó. Este processo é iterativo, até atingir o ponto em que o resultado do nó são apenas ramificações que contêm só elementos com o mesmo tipo de saídas, que são chamadas de *folhas* ou *nós foliares* (ALPAYDIN, 2010).

Neste trabalho é usada uma DT binária. Como os dados de entrada são numéricos, a função $f_m(x)$ é uma comparação entre os valores do conjunto de dados que entra no nó e o limiar determinado w_{m0} dado pela Equação (5.1).

⁷ O fato de ser simples de implementar não garante o sucesso do algoritmo.

$$f_m(x): x_j > w_{m0} \quad (5.1)$$

O nó de decisão divide as entradas em dois espaços, este processo é chamado de *divisão binária*. Neste tipo de método, a qualidade da divisão é avaliada usando medidas de impureza. Há várias funções para medir impurezas (denotadas por ϕ), as quais, no caso de se ter apenas duas classes, devem obedecer a uma série de propriedades propostas em (DEVROYE; GYÖRFI; LUGOSI, 1996; ALPAYDIN, 2010). A Equação (5.2) descreve o Índice de Diversidade de Gini (GDI) proposto em (BRIEMAN *et al.*, 1984), o qual é a função usada para medir a impureza dos nós utilizados, sendo p^i a probabilidade de se obter a classe i e K é o total de classes no conjunto de saída.

$$\phi(p^i) = 1 - \sum_{i=1}^K (p^i)^2 \quad (5.2)$$

O algoritmo usado para implementar a DT foi a Árvore de Classificação e Regressão (CART) (BRIEMAN *et al.*, 1984), que se baseia na escolha da divisão que causa a maior queda de impureza, que representa a diferença entre a impureza do conjunto de dados que entra no nó de decisão m (Equação (5.2)) e a impureza total de cada uma das ramificações que saem do nó, expressa na Equação (5.3) (ALPAYDIN, 2010; IBM, 2019).

$$\phi(p_m^i)' = \sum_{j=1}^n \frac{N_{mj}}{N_m} \left(1 - \sum_{i=1}^K (p_{mj}^i)^2 \right) \quad (5.3)$$

O parâmetro N_{mj} é a quantidade de elementos catalogados na saída j do nó de decisão m , N_m é a quantidade dos elementos que entram no nó de decisão m , p_{mj}^i é a probabilidade de se escolher um elemento da classe i na ramificação j saindo do nó de decisão m .

O algoritmo *CART* possui as seguintes características, que são entendidas como vantagens para este trabalho (NISBET; ELDER; MINER, 2009):

- Como é um algoritmo não paramétrico, os dados não precisam pertencer a uma determinada distribuição;
- Não é afetado significativamente pelos *outliers* na entrada;
- Incorpora a verificação dos erros com o conjunto de treinamento e com a validação cruzada, acrescentando a precisão; e
- Pode usar as variáveis de entrada mais de uma vez em diferentes partes da árvore, aumentando a capacidade de resolver complexidades entre as variáveis de entrada e saída.

5.1.2 Redes Neurais Artificiais

As redes neurais artificiais (ANN) são um modelo computacional que simula a estrutura do cérebro humano. Foram propostas pela primeira vez por (MCCULLOCH; PITTS, 1943) e se baseiam na representação de neurônios e suas interações de modo semelhante ao que ocorre no cérebro. A Figura 5-2 representa o modelo de um *neurônio*, que é a pedra angular deste método. Ele tem *sinapses*, que é um elemento que conecta as entradas com o corpo do neurônio; as quais são caracterizadas por um *peso* w_k que pode ser positivo ou negativo, algo que não acontece no cérebro real. Ao corpo do neurônio, conhecido como *somador*, chega a informação proveniente das sinapses ou o que é o mesmo: a multiplicação do sinal x_i na entrada das sinapses i multiplicada pelo seu peso w_{ki} . No corpo, são adicionadas todas as informações recebidas das diferentes sinapses e os *bias* b_k . O resultado desta soma é a entrada do último elemento que compõe o neurônio, chamado de *função de ativação* φ , que determina a saída do neurônio. Existem diferentes tipos de funções de ativação que possuem funcionalidades específicas e, portanto, sua escolha correta determina fortemente o resultado final (HAYKIN, 2009).

Figura 5-2- Modelo matemático de um neurônio artificial.

Fonte: (HAYKIN, 2009).

Do ponto de vista matemático, um neurônio k é representável pelas Equações (5.4) e (5.5), que descrevem o corpo do neurônio e a função de ativação, respectivamente.

$$v_k = \sum_{j=1}^m w_{kj} x_j + b_k \quad (5.4)$$

$$y_k = \varphi(v_k) \quad (5.5)$$

Os parâmetros $x_1, x_2, \dots, x_m$ são as entradas; $w_{k1}, w_{k2}, \dots, w_{km}$ são os pesos de cada uma das sinapses; b_k é o *bias*; $\varphi(\cdot)$ é a função de ativação; y_k é a saída do neurônio e v_k é o campo local induzido do neurônio (HAYKIN, 2009).

Quando (MCCULLOCH; PITTS, 1943) propuseram este modelo de neurônio, eles estavam cientes das suas limitações ao lidar com problemas não linearmente separáveis. Eles podem representar com precisão apenas os tipos de função OR, AND e NOT. Para resolver problemas mais complexos e não linearmente separáveis, são usadas redes neurais, que são um agrupamento de neurônios organizados em diferentes camadas, nas quais a saída de cada uma delas está associada à entrada de cada um dos neurônios da próxima camada (RUSSELL; NORVIG; EDWARDS, 2010). Há várias estruturas para organizar os neurônios, uma das mais populares e que é usada neste trabalho é intitulada Multicamadas Perceptron (*Perceptron Multilayer*). A função de ativação usada nesta estrutura é vista na Equação (5.6), denominada função sigmoide (ou *logistic function*). Ela é usada principalmente em tarefas de classificação e uma das suas vantagens é seu caráter diferenciável, elemento que é importante para o processo de treinamento do algoritmo.

$$\varphi(v_k) = \frac{1}{1 + e^{-av_k}} \quad (5.6)$$

O parâmetro a é a inclinação da sigmoide e v_k é a combinação linear das entradas e saídas. O processo de aprendizagem se baseia em minimizar o erro entre a saída da ANN e a saída desejada. Esta diferença, chamada função custo (J), é calculada usando o *MSE*; como mostrado na Equação (5.7), onde $c_k(n)$ é o n -ésimo valor do vetor c que contém as saídas desejadas; $y_k(n)$ é a saída do sistema, produto dos n -ésimos valores do vetor x que contém os valores de entrada (ALPAYDIN, 2010).

$$J = \|c_k(n) - y_k(n)\|^2 \quad (5.7)$$

Para encontrar os valores que minimizam J , o método do *gradient descent* é usado, no qual os pesos w_{kj} são atualizados usando a Equação (5.8), a qual é intitulada *generalized delta rule* (HAYKIN, 2009; RUSSELL; NORVIG; EDWARDS, 2010).

$$w_{ki} = \lambda w'_{ki} + \eta \delta_j(n) y_k(n) \quad (5.8)$$

O parâmetro w'_{ki} é o valor anterior dos pesos w_{ki} ; w_{ki} é o valor atualizado dos pesos das sinapses i do neurônio k ; η é uma constante conhecida como taxa de aprendizado, quanto maior ela for mais rápida é a convergência⁸, embora se for muito grande, pode torná-la instável; λ é a constante do momento. Outras variáveis são os gradientes locais dos neurônios j e k , δ_j e δ_k , respectivamente. As Equações (5.9) e (5.10) são baseadas nas fórmulas descritas pelo algoritmo de *Back-propagation*, para calcular δ_j e δ_k . A Figura 5-3 ilustra a estrutura da rede treinada com este método.

⁸ Isto ocorre pois a distância que pode ser percorrida em cada iteração é diretamente proporcional a η , portanto, o número de iterações necessárias para atingir o mínimo da função custo diminui com o aumento de η .

$$\delta_k(n) = (c_k(n) - y_k(n)) \frac{d\varphi(v_k(n))}{dv_k(n)} \quad (5.9)$$

$$\delta_j(n) = \frac{d\varphi(v_j(n))}{dv_j(n)} \sum_k \delta_k(n) w_{kj}(n) \quad (5.10)$$

Figura 5-3- Gráfico de fluxo de sinais destacando os detalhes do neurônio da saída k conectado ao neurônio oculto j .

Fonte: (HAYKIN, 2009).

As ANN treinadas usando o método de *Back-propagation* provaram ser um estimador universal capaz de lidar com não linearidades. Sua estrutura permite o processamento paralelo e trabalha com uma grande quantidade de dados. Sua capacidade de distinguir padrões de comportamento dos dados entre grandes conjuntos deles é outro recurso que pode ser útil neste trabalho (SUYKENS; DE MOOR; VANDEWALLW, 1996).

5.1.3 Máquina de Vetores de Suporte

As Máquinas de Vetores de Suporte (SVM) são um método de aprendizado supervisionado que cria um separador com a maior margem possível (Figura 5-4). A base deste mecanismo de classificação é selecionar os pontos do conjunto de treinamento que definam uma separação entre as diferentes classes contidas no conjunto. Estes pontos localizados no limite dos conjuntos são chamados de máquina de vetores de suporte. No caso em que o conjunto de dados de entrada não seja linearmente separável, várias soluções são usadas para separá-los, uma delas é chamada de *kernels*, que são funções de similaridade usadas para representar os dados de uma dimensão d em outra dimensão z , na qual eles são linearmente separáveis (ALPAYDIN, 2010).

Figura 5-4- Arquitetura de uma SVM usando uma rede de base radial.

Fonte: (HAYKIN, 2009).

O problema a ser resolvido pela SVM é apresentado na Equação (5.11), na qual se procura a maior separação possível entre as classes, minimizando os valores dos pesos w , sendo x^t as entradas do sistema, r^t são as saídas desejadas do sistema, $t = 1, 2, 3, \dots, N$ é o número de amostras que possuem o conjunto de treinamento e $w_0 = b$ é o *bias* (ALPAYDIN, 2010; RUSSELL; NORVIG; EDWARDS, 2010).

$$\frac{1}{2} \|w\|^2 \text{ restricta a } r^t(w^T x^t + w_0) \geq +1, \forall t \quad (5.11)$$

A Equação (5.11) é resolvida com o operador de Lagrange (α^t são multiplicadores de Lagrange) como observado na Equação (5.12). Esta deve ser minimizada em relação a w_0 e w , e deve ser maximizada em relação às $\alpha^t \geq 0$; condições mostradas nas Equações (5.13) e (5.14) (ALPAYDIN, 2010).

$$L_p = \frac{1}{2} \|w\|^2 - \sum_t \alpha^t r^t (w^T x^t + w_0) + \sum_t \alpha^t \quad (5.12)$$

$$\frac{\partial L_p}{\partial w} = 0 \rightarrow w = \sum_t \alpha^t r^t x^t \quad (5.13)$$

$$\frac{\partial L_p}{\partial w_0} = 0 \rightarrow \sum_t \alpha^t r^t = 0 \quad (5.14)$$

A solução anterior é aceita para sistemas linearmente separáveis; para os restantes não linearmente separáveis, as funções *kernel* ($K(x^t, x)$) são usadas. Há várias destas funções, nesta dissertação o *kernel* gaussiano é usado, o qual é capaz de lidar com a maioria das não linearidades; seu uso nas funções de classificação pode ser observado em vários trabalhos (WANG, 2005). A Equação (5.15) descreve o *kernel* gaussiano, sendo o parâmetro s geralmente definido pelo usuário.

$$K(x^t, x) = \exp \left[\frac{\|x^t - x\|^2}{2s^2} \right] \quad (5.15)$$

Finalmente, para ter uma representação do problema matemático, a Equação (5.16) é apresentada. Este é o resultado da substituição das Equações (5.13), (5.14) e (5.15) na Equação (5.12) e do uso da função *kernel* selecionada (ALPAYDIN, 2010).

$$L_p = \sum_t \alpha^t - \frac{1}{2} \sum_t \sum_s \alpha^t r^t \alpha^s r^s K(x^t, x^s) \quad (5.16)$$

Ao encontrar os valores de t para os quais α^t é maior que zero, são encontrados os valores limite que definirão o SVM. Eles são usados para gerar um limiar entre as diferentes classes e definir a possível classificação das saídas do algoritmo usando a Equação (5.19). Nesta, w_m são os pesos do sistema; x_m são as entradas do sistema; m é o número de entradas; $K(x, x_m)$ é o *kernel*; b é o *bias* e y é a saída que pode ser interpretada como: quando for maior que zero pertence a uma classe e quando for menor que zero pertence a outra (RUSSELL; NORVIG; EDWARDS, 2010).

$$y = \sum_m w_m K(x, x_m) + b \quad (5.17)$$

Ao contrário dos modelos de ANN, a SVM fornece uma solução única. Além disso, como se destina a encontrar o separador que melhor descreve o conjunto de dados, é uma excelente solução em tarefas de classificação em que os dados são desequilibrados. Em muitas aplicações de ML, quando se compara o desempenho da ANN com a SVM, a SVM tem um desempenho melhor; mas tem um alto custo computacional e sua operação correta depende muito do *kernel* escolhido (AURIA; MORO, 2008).

5.2 Etapa de obtenção e preparação dos dados para o treinamento

Nos projetos de IA obter os dados para treinamento é considerado o primeiro passo na construção de um algoritmo de ML. O trabalho com aprendizado supervisionado, exige dados de entradas e saídas. Esses dados puros, comumente carecem de sentido para o algoritmo, por isso eles precisam ser preparados. Nesta etapa as duas ações fundamentais são analisar e catalogar esses dados. Estes conjuntos de dados provêm do resultado de testes de validação em experimentos de identificação e de resultados de KPIs em testes de avaliação de malhas de controle. Especificamente, foram realizados três testes:

- 1) Teste nº 1: Consiste em estimular as plantas configuradas em malha fechada com uma série de sinais de excitação e, em seguida, coletar os dados de entrada e saí-

da do sistema. O objetivo deste experimento é ter dados suficientes para realizar a identificação e obter um grupo variado de modelos com resultados díspares, que representem o sistema de forma correta e incorreta. Dois tipos de sinais foram usados: degrau e GBN. O número de testes muda em cada planta. Estes foram desenvolvidos nos ambientes simulados explicados nos Apêndice A e Apêndice B.

- 2) Teste nº 2: Similar ao Teste nº 1, mas neste caso são excitados os modelos identificados visando usar os dados salvos para validar os modelos, usando os métodos descritos no Capítulo 3. Os dados observados usados para validar são boas representações dos processos, no caso do ambiente simulado da planta de pH são os dados obtidos nos experimentos de identificação da Seção 4.2.
- 3) Teste nº 3: Consiste em colocar no *setpoint* das plantas uma série de degraus. As plantas estarão configuradas em malha fechada e controlada por um MPC, que mantém os mesmos pesos de sintonia e parâmetros de configuração. O objetivo deste teste é comparar o desempenho da malha de controle ao usar os diferentes modelos identificados no Teste nº 1 e obter seus respectivos KPIs. A sintonia do MPC foi baseada em tentativa e erro. Os pesos da sintonia do MPC da planta de pH foram encontrados em (PÉREZ SERRANO, 2019).

5.2.1 Análise dos dados de treinamento

Na etapa de análise dos dados é verificada a sua composição e distribuição, buscando que forneçam o maior número possível de informações sobre a situação estudada. Comportamentos que devem ser evitados para se ter um bom conjunto de dados para treinamento são os seguintes:

- Valores de entradas iguais que forneçam saídas distintas (alta entropia);
- Conjunto de dados com baixa resolução, isto é, para variações muito pequenas nas entradas, mudanças aleatórias na saída;
- Conjunto de dados desbalanceados, onde a quantidade de exemplos que descreve um comportamento do sistema é muito maior do que o resto dos dados que descrevem os outros comportamentos;
- Conjunto de dados pequenos que não descrevam bem a situação de interesse;
- Os grandes *outliers* também dificultam o processo de treinamento, pois eles complicam a etapa de normalização dos dados.

Em resumo, um bom conjunto de dados para treinamento é aquele no qual os dados estejam bem espalhados, onde a quantidade de exemplos que descreva um comportamento seja balanceada e onde os dados permitam distinguir os distintos comportamentos do sistema.

5.2.1.1 Análise dos Testes de Hipóteses

Os resultados dos testes de hipóteses não foram os esperados. Na Figura 5-5 se mostra um *boxplot* com os resultados dos distintos métodos de validação baseados em teste de hipóteses dos 3150 modelos obtidos⁹ na etapa de identificação da planta de pH. Tanto no caso (a), onde são usados os dados processados como no caso (b) onde não, pode-se observar um comportamento similar.

Figura 5-5- Distribuição dos resultados dos testes de hipóteses na Planta Piloto de Neutralização de pH.

Fonte: AUTOR.

Olhando a Figura 5-5 é possível intuir que nenhum destes testes gera informações significativas sobre a qualidade dos modelos. Os testes de Kolmogorov–Smirnov (K-S), Anderson–Darling (A-D) e Shapiro–Wilk (S-W) têm média zero, onde o 1º e o 3º quartis são também valores aproximadamente zero e os desvios padrão são $1,76e^{-87}$, $4,25e^{-5}$ e 0, respectivamente. Estas métricas fornecem um mesmo resultado (os erros de predição não têm uma distribuição normal) para um conjunto bem variado de respostas dos modelos. Baseado nestes dados, pode-se concluir que têm uma alta entropia. O teste de Wilcoxon foi capaz discriminar melhor os modelos, mas ainda assim as respostas têm uma distribuição não desejável, com muitos valores em uma faixa bem pequena.

⁹ Essa quantidade de modelos é produto da combinação das possíveis ordens dos polinômios. A explicação de como foram obtidos os modelos foi detalhada no Apêndice B.

A literatura indica que este comportamento pode ocorrer quando se lida com quantidades grandes de amostras (LIU *et al.*, 2011). Outro fator poderia ser a alta sensibilidade que têm estes testes aos dados com valores repetidos, excetuando o teste de Wilcoxon, fruto de arredondamento ou imprecisão na medição, o que faz com que as hipóteses sejam rejeitadas frequentemente (ROYSTON, 1992). As Figura 5-6 e Figura 5-7 exibem a CDF para dois modelos distintos, que representam as dinâmicas mais comuns nas malhas da Planta Piloto de Neutralização de pH. Esses quatro modelos foram rejeitados por todos os testes de hipóteses.

Figura 5-6- Resposta dos modelos n° 19 das duas malhas da Planta Piloto de Neutralização de pH e suas respectivas funções CDF.

Fonte: AUTOR.

As CDF dos dados reais (CDF_PV_Real) e dos estimados (CDF_PV_Estimada) são muito similares, ambas confirmam que há uma quantidade grande de valores repetidos nos dois conjuntos de dados. No caso dos modelos preditos, este fenômeno era esperado, pelo fato de se estar identificando um sistema de malha fechada, os quais em teoria devem estabilizar em um determinado valor, mas no caso dos dados reais não era esperado um comportamento tão similar com os modelos identificados. Esta situação faz com que a CDF dos resíduos de validação tenha uma inclinação muito maior do que a CDF normal na faixa dos valores perto da média (desvio padrão igual 0). Este fenômeno se acentua em modelos “melhores¹⁰”, como acontece no Modelo n°19 (Figura 5-6). A diferença na inclinação se traduz em um maior distanciamento entre as CDF da normal e dos resíduos, portanto a porcentagem de rejeição destes testes foi quase de 100%.

¹⁰ Esta etapa é baseada só em análises visuais; entende-se que o “melhor” modelo é aquele que seja similar ao sistema estável linear ideal.

Figura 5-7- Resposta dos modelos n° 2 das duas malhas da Planta Piloto de Neutralização de pH e suas respectivas funções CDF.

Fonte: AUTOR.

Para tentar reduzir a taxa de rejeições e obter o conjunto de dados de validação com menor entropia, se reduziu o número de dados dos experimentos de validação, mudando o tempo de amostragem para 60 s. Isto diminui o número de dados a 248 e reduz o número de valores repetidos. O resultado deste novo conjunto de dados é visto na Figura 5-8.

Figura 5-8- Distribuição dos resultados dos testes de hipóteses com tempo de amostragem de 60 s.

Fonte: AUTOR.

A Tabela 5.1 exhibe dados estatísticos que resumem os gráficos da Figura 5-8. Usar menos amostras para validar reduziu a taxa de rejeições e diminuiu a entropia. Ainda assim, as mudanças foram insuficientes. Ademais, aumentar o tempo de amostragem gera perda de informações entre o modelo e o sistema identificado, criando novas incertezas.

Tabela 5.1- Estatística dos resultados dos testes de hipóteses da Planta Piloto de Neutralização de pH.

	K-S		A-D		S-W		Wilcoxon	
	Nível	pH	Nível	pH	Nível	pH	Nível	pH
Média	0.0013	0.0004	0.0006	0.0005	0	0	0.0119	0.0033
Desvio padrão	0.0209	0.0074	0.0045	0.0001	0.0019	0.0002	0.0867	0.0458
Limite superior	0.8443	0.2479	0.2546	0.0029	0.1076	0.012	0.9804	0.9626
Limite inferior	0	0	0	0	0	0	0	0
Modelos rejeitados (%)	99.4	99.71	99.97	100	99.97	100	97.78	99.59
Modelos aceitos (%)	0.6032	0.2857	0.0317	0	0.03	0	2.22	0.41

Fonte: AUTOR.

A Figura 5-9 e a Tabela 5.2 mostram a resposta dos modelos identificados da malha de nível e os resultados de seus testes de hipóteses. Só o modelo n° 2924 conseguiu ser aceito por vários dos testes, mas o seu comportamento não é muito similar ao de uma malha fechada ideal. Destaca-se o caso das respostas dos modelos n° 19 e n° 2045 que são quase idênticas, mas os resultados do teste de Wilcoxon têm valores muito diferentes. Curiosamente, respostas similares ao do modelo n° 747 foram aceitas pelo teste K-S.

Figura 5-9- Comparação das respostas dos modelos números 2045, 19, 2949, 2 e 747 da malha de nível.

Fonte: AUTOR.

Tabela 5.2- Resultado dos testes de hipóteses dos modelos números 2924, 2, 2045, 19 e 747 da malha de nível.

Modelo No.	p-value			
	K-S	A-D	S-W	Wilcoxon
2924	0.8443	0.2546	0.1076	0
2	0	0.0005	0	0
2045	0	0.0005	0	0.8477
19	0	0.0005	0	0.0067
747	0.0905	0.0005	0	0

Fonte: AUTOR.

Fazendo uma análise mais detalhada do comportamento do teste de Wilcoxon, na Figura 5-10 são mostrados quatro modelos quase idênticos, mas seus resultados utilizando o teste de Wilcoxon variam muito (vide a Tabela 5.3). Este comportamento não é desejável, pois neste caso mudanças quase nulas nos modelos provocam mudanças não proporcionais na saída e podem dificultar o processo de treinamento.

Figura 5-10- Comparação das respostas dos modelos números 3044, 3056, 3041 e 3035 da malha de nível.

Fonte: AUTOR.

Tabela 5.3- Resultado dos testes de hipóteses dos modelos números 3044, 3056, 3041 e 3035 da malha de nível.

Modelo No.	K-S	A-D	p-value	
			S-W	Wilcoxon
3044	2.685e ⁻⁸	0.0005	1.332e ⁻¹⁵	0.4441
3056	2.499e ⁻⁸	0.0005	1.554e ⁻¹⁵	0.2669
3041	2.704e ⁻⁸	0.0005	1.554e ⁻¹⁵	0.8190
3035	2.499e ⁻⁸	0.0005	1.554e ⁻¹⁵	0.7047
3044	2.685e ⁻⁸	0.0005	1.332e ⁻¹⁵	0.4441

Fonte: AUTOR.

Planta de Clarke

Na planta simulada de Clarke foi repetido o mesmo procedimento da planta de pH, com a diferença que os dados não precisaram ser tratados, pois o índice SNR não é muito baixo (12,56 dB) e o ruído foi gerado de modo artificial, com uma distribuição normal. A Figura 5-11 mostra uma comparação entre o resultado de aplicar os testes de hipóteses para validar os 5250 modelos estimados. Neste caso, os *boxplots* com intervalo de amostragem $T_a=1s$ são o conjunto de validação com 10501 dados e $T_a=42s$ são o conjunto de validação com 251 dados.

Figura 5-11- Distribuição dos resultados dos testes de hipóteses na planta de Clarke.

Fonte: AUTOR.

Tabela 5.4- Estatísticas dos resultados dos testes de hipóteses da planta de Clarke.

	KS		AD		SW		Wilcoxon	
	Ta=1	T=42s	Ta=1	T=42s	Ta=1	T=42s	Ta=1	T=42s
Média	0	0.055	0.00060	0.0046	0	0.0018	0.2049	0.6545
Desvio padrão	0	0.157	0.0054	0.0131	0.0003	0.0144	0.1088	0.2127
Limite superior	0	0.842	0.3381	0.15	0.0189	0.2853	0.9468	1
Limite inferior	0	0	0	0	0	0	0	0
Modelos rejeitos (%)	100	87.45	99.92	97.16	100	98.96	7.68	7.47
Modelos aceitos (%)	0	12.55	0.08	2.84	0	1.04	92.32	92.53

Fonte: AUTOR.

Os resultados dos testes K-S, A-D e S-W obtidos nesta planta concordam em grande parte com os da planta de pH (vide a Tabela 5.4), com a particularidade que a porcentagem de modelos aceitos foi maior, assim como a quantidade de modelos aceitos por vários testes (45). Os resultados do teste de Wilcoxon são diferentes, ele aceitou a maioria dos modelos. Este comportamento é associado a dois fatores: o ruído usado é gaussiano e com uma variabilidade menor que na planta de pH, e o cálculo de Wilcoxon não é afetado pelos dados com valores repetidos. Seus resultados mostram uma melhor distribuição. Mas pelo fato de ter o mesmo comportamento observado com os modelos números 3044, 3056, 3041 e 3035 da malha de nível, os resultados do teste de Wilcoxon não seriam de muita ajuda no treinamento.

Na Figura 5-12 exibem-se as respostas de modelos identificados (PV_Estimada) e da planta simulada (PV) no lado esquerdo e na direita as funções de CDF do modelo identificado (PV_Estimada), da planta (PV_Real), do resíduo (Resíduo) e da distribuição normal (Normal). Neste caso, as respostas dos modelos 22 e 2179 são quase idênticas, mas o modelo 2179 é aceito por todos os testes, enquanto o modelo 22 é rejeitado por to-

dos os testes (vide a Tabela 5.5). Os cinco modelos têm uma dinâmica bem similar, mas os resultados dos testes não. Os resultados do K-S e Wilcoxon têm uma maior concordância com a análise visual. Mas no caso K-S deve-se lembrar que seu poder de decisão é o menor dos testes de hipóteses baseados em diferença de CDF (RAZALI; WAH, 2011).

Figura 5-12- Resposta dos modelos números 16, 1841, 1612, 22 e 2179 da planta de Clarke e suas respectivas CDF.

Fonte: AUTOR.

Tabela 5.5- Resultados dos testes de hipóteses dos modelos números 16, 1841, 1612, 22 e 2179 da planta de Clarke.

Modelo No.	KS		AD		SW		Wilcoxon	
	Ta=1 s	T=42 s	Ta=1 s	T=42 s	Ta=1 s	T=42 s	Ta=1 s	T=42 s
16	0	0.4514	0.0005	0.0184	0	0	0.2632	0.6206
1841	0	0.6456	0.0005	0.0821	0	0.0024	0.1469	0.5255
1612	0	0.7577	0.0005	0.0916	0	0.195	0.2637	0.7493
22	0	0	0.0005	0.0232	0	0.0013	0.269	0.5981
2179	0	0.61690	0.0005	0.0911	0	0.1564	0.2683	0.6347

Fonte: AUTOR.

Baseado nos resultados de ambas as plantas, são feitas as seguintes observações sobre os testes de hipóteses:

- Têm fraco desempenho com conjuntos com muitos dados. O número ótimo de dados para fazer estes testes é bem menor do que o número comum de dados dos conjuntos de validação na área de Controle de Processos (LIU *et al.*, 2011).
- Quando se trabalha em malha fechada com modelos e planta estáveis, o resultado vai ser influenciado principalmente pela distribuição do ruído, não pela semelhança entre as CDF do modelo e da planta. Portanto, os dados reais com ruído colorido aumentam a probabilidade de rejeição, como acontece nos casos dos modelos números 2 e 19 da planta de Clarke.
- Valores repetidos afetam o desempenho das métricas K-S, A-D e S-W porque a CDF dos modelos vai ter um comportamento distante da distribuição normal em algumas faixas, principalmente no caso de modelos estáveis, nos quais essa faixa é próxima da média ($\sigma=0$). Assim, o desempenho não melhora usando dados de plantas estáveis em malha fechada com menor ruído, como ocorre na planta de Clarke.
- Não geram informação sobre a dinâmica do problema e fornecem pouca informação para diferenças entre as respostas dos modelos.
- Para os modelos aceitos a robustez destes resultados é pequena, principalmente o teste de Wilcoxon, visto que pequenas variações na dinâmica dos modelos provocam grandes mudanças no resultado.

5.2.1.2 Análises das Métricas Adimensionais da Média do Erro

A distribuição dos resultados das métricas adimensionais da média do erro (MAME) é mostrada na Figura 5-13 (a) e (b). O comportamento e a distribuição deste conjunto de dados foram os esperados. Neste caso, excetuando-se os resultados de Anova, o resto dos resultados têm uma menor entropia e resolução do que aqueles dos testes de hipóteses. O comportamento de Anova é muito similar com os testes de hipóteses pois seu resultado final é uma probabilidade calculada baseada no teste estatístico t, daí este comportamento. Segundo os resultados das MAME e os critérios estabelecidos no Capítulo 3, a grande maioria de modelos é aceita (vide as Tabela 5.6 e 5.7).

Tabela 5.6- Estatísticas dos resultados das MAME na Planta Piloto de Neutralização de pH com os dados sem processar.

	FIT		TIC		d'		C _r		C _{s&G}		T _A	
	H	pH	H	pH	H	pH	H	pH	H	pH	H	pH
Média	-1.5e ⁺²⁸⁵	-1.3e ⁺²⁵⁰	0.1223	0.0726	-0.1886	-0.07	0.8623	1.683	1.6e ⁺¹²⁶	3.1e ⁺¹⁴⁰	0.9897	0.9957
Desvio padrão	Inf	Inf	0.271	0.1794	0.6964	0.7025	6.693	11.11	8.8e ⁺¹²⁷	1.7 ⁺¹⁴²	0.090	0.058
Mediana	-0.4602	-0.1703	0.019	0.0176	-0.4326	-0.1809	0.0279	0.0259	0.0385	0.34348	1	1
Limite superior	0.8507	0.8613	1	1	0.8962	0.9194	114.9	127.6	4.9e ⁺¹²⁹	9.8e ⁺¹⁴³	1	1
Limite inferior	-4.9e ⁺²⁸⁸	-4.1e ⁺²⁵³	0.002	0.0021	-1	-1	0	0	0.0013	0.0013	0	0
Modelos rejeitos (%)	75.31	70.61	9.78	3.59	75.79	70.04	15	5.44	12.63	4.99	1.87	0.76
Modelos aceitos (%)	24.69	29.39	90.22	96.41	24.21	29.96	85	94.56	87.37	95.01	98.13	99.24

Fonte: AUTOR.

Tabela 5.7- Estatísticas dos resultados das MAME na Planta Piloto de Neutralização de pH com os dados processados.

	FIT		TIC		d'		C _r		C _{s&G}		T _A	
	H	pH	H	pH	H	pH	H	pH	H	pH	H	pH
Média	-1.5e ⁺²⁸⁵	-1.3e ⁺²⁵⁰	0.1222	0.0726	-0.1941	-0.0726	0.8623	1.683	1.6e ⁺¹²⁶	3.1e ⁺¹⁴⁰	0.9899	0.9957
Desvio padrão	Inf	Inf	0.2711	0.1794	0.6993	0.7032	6.693	11.11	8.8e ⁺¹²⁷	1.7e ⁺¹⁴²	0.0902	0.0578
Mediana	-0.4638	-0.1691	0.0017	0.0176	-0.4469	-0.1863	0.0279	0.0258	0.0387	0.0343	1	1
Limite superior	0.8704	0.8645	1	1	0.9063	0.9231	114.9	127.6	4.9 e ⁺¹²⁹	9.8e ⁺¹⁴³	1	1
Limite inferior	-4.9e ⁺²⁸⁸	-4.1e ⁺²⁵³	0.0017	0.002	-1	-1	2e ⁻⁵	9e ⁻⁵	0.00110	0.00130	0	0
Modelos rejeitos (%)	75.25	70.49	9.78	3.59	75.86	70.07	15	5.44	12.63	4.99	1.68	0.73
Modelos aceitos (%)	24.75	29.51	90.22	96.41	24.14	29.93	85	94.56	87.37	95.01	98.32	99.27

Fonte: AUTOR.

Figura 5-13- Distribuição dos resultados das MAME na Planta Piloto de Neutralização de pH (a) dados de validação sem processar (b) dados processados

Fonte: AUTOR.

As Tabela 5.6 e Tabela 5.7 confirmam que o tratamento dos dados feito neste trabalho não introduz uma variação significativa na distribuição dos resultados da validação. Neste caso, a principal mudança ao processar os dados é que aumentou a proximidade da resposta simulada com relação à resposta medida. Sendo assim, decidiu-se usar os dados processados para o resto das análises.

Na Figura 5-14 mostra-se a resposta dos modelos números 3117, 1930 e 1932 da malha de nível. Os resultados da Tabela 5.8 confirmam que o ganho dos modelos influencia muito o resultado destes métodos de validação, mas não mostram que sejam muito sensíveis a outras mudanças nas características dinâmicas da resposta do modelo.

Figura 5-14- Respostas dos modelos números 3117, 1930 e 1932 da malha de nível.

Fonte: AUTOR.

Tabela 5.8- Resultados das MAME dos modelos números 3117, 1930 e 1932 da malha de nível.

Model No.	FIT	TIC	d'	C _r	C _{s&G}	T _A	Ganho / (e _{ss})	t _r (s)	t _s (s)	M _p
3117	-0.0545	0.0138	-0.231	0.0205	0.0278	1	1.02 / (1.81)	95	417	10.84%
1930	0.8552	0.0019	0.8866	0.0008	0.002	1	1.002 / (0.16)	109	308	12.85%
1932	0.6687	0.0044	0.6059	0.0061	0.0082	1	0.99 / (0.55)	89s	306	12.09%

Fonte: AUTOR.

Na Figura 5-15 exibe-se a resposta de dois modelos identificados da malha de pH. Neste caso, baseado em uma análise visual, nota-se que os dois modelos são bem distintos e não têm muita similitude entre eles, nem com a planta. Mas a informação mostrada na Tabela 5.9 dá uma ideia distinta. Segundo esta e os valores fixados no Capítulo 3, as duas respostas têm uma similitude aceitável com a planta. Os resultados confirmam que nesta situação estes métodos não são muito influenciados por outras características da dinâmica além do ganho.

Tabela 5.9- Resultados das MAME dos modelos números 1446 e 1467 da malha de pH.

Model No.	FIT	TIC	d'	C _r	C _{s&G}	T _A	Ganho / (e _{ss})	t _r (s)	t _s (s)	M _p
1446	0.6209	0.0056	0.6523	0.0071	0.0098	1	0.99 / (0.0679)	64	555	4.93%
1467	0.6123	0.0058	0.6747	0.0063	0.009	1	0.991 / (0.0635)	31	404	48.18%

Fonte: AUTOR.

Figura 5-15- Respostas dos modelos números 1446 e 1467 da malha de pH.

Fonte: AUTOR.

Planta de Clarke

A Figura 5-16 mostra a distribuição dos resultados para a planta de Clarke, os quais têm algumas semelhanças com o comportamento observado na planta de pH, ainda que

o ruído desta seja bem menor que da planta de pH. Trabalhar sem ruído melhora a confiabilidade da validação, mas neste caso causa uma diminuição na faixa onde estão distribuídos os dados. O conjunto de dados com os resultados das MAME nesta planta têm o primeiro e o terceiro quartis muito mais perto que na planta de pH (vide a Tabela 5.10). Isto poderia ser uma dificuldade na hora de treinar os algoritmos, se esta redução significa uma quantidade alta de valores repetidos que não correspondem a exemplos iguais.

Figura 5-16- Distribuição dos resultados das MAME na planta de Clarke.

Fonte: AUTOR.

Tabela 5.10- Estatísticas dos resultados das MAME da planta de Clarke.

	FIT	TIC	d'	C_r	C_{s&G}	T_A
Média	-2.2e+293	0.1781	0.7926	2.0557	4.8e+147	0.7122
Desvio padrão	Inf	0.3023	0.4426	12.2106	3.4e+149	0.1598
Mediana	0.9133	0.0457	0.9405	0.0213	0.0316	0.7585
Limite superior	0.9333	1	0.945	132	2.4e+151	1
Limite inferior	-1.1e+297	0.0334	-1	0.0012	0.0213	0
Modelos rejeitos (%)	13.13	13.84	9.72	82.98	85.66	95.43
Modelos aceitos (%)	86.87	86.16	90.28	17.02	14.34	4.57

Fonte: AUTOR.

A Figura 5-17 apresenta as repostas de três modelos identificados. Segundo uma análise visual, o melhor modelo e o único que descreve a dinâmica do processo é o modelo número 2443 (amarelo). Mas olhando os resultados das MAME na Tabela 5.11, todos os modelos são uma boa representação do processo. A Tabela 5.11 apresenta também os valores da resposta transitória dos modelos, confirmando diferenças significativas com a planta.

Figura 5-17- Respostas dos modelos números 2443, 4669 e 4720 da planta de Clarke.

Fonte: AUTOR.

Tabela 5.11- Resultados das MAME dos modelos números 2443, 4669 e 4720 da planta de Clarke.

Model No.	FIT	TIC	d'	C _r	C _{s&G}	T _A	Ganho / (e _{ss})	t _r (s)	t _s (s)	M _p
2443	0.9296	0.0354	0.9434	0.0182	0.0244	0.797	0.9901 / (9.9e ⁻³)	6	29	16.34%
4669	0.8969	0.0518	0.9327	0.0281	0.0344	0.7089	0.9895 / (1.05e ⁻²)	11	217	29.76%
4720	0.8949	0.0528	0.9274	0.0288	0.0348	0.7055	0.9874 / (1.26e ⁻²)	10	480	25.37%

Fonte: AUTOR.

Baseado nas informações de ambas as plantas são feitas as seguintes observações:

- Todas as métricas MAME são fortemente influenciadas pelas diferenças de ganho entre os modelos identificados e os dados de validação. Nos casos de Russell e Sprague & Gueers, analisando-se a fase não melhora este comportamento. Portanto, não dão uma imagem clara das diferentes dinâmicas.
- O denominador do TIC leva em conta os resultados preditos, portanto a sua sensibilidade ante as mudanças bruscas dos valores preditos é menor. Consequentemente, não se julga que empregar o valor de 0,4 como limiar para aceitar o modelo seja bom nestas aplicações.
- A interpretação destas métricas é mais lógica quando se trabalha com menos ruído. Por isto, precisa-se de IA para atingir uma melhor interpretação.
- Os resultados destas métricas apresentam uma boa coerência, mas a característica do FIT, Russell e Sprague & Gueer de não ter limite pode ser uma desvantagem na hora de treinar os algoritmos de inteligência artificial.
- A alta porcentagem de aceito dos modelos parece indicar que os critérios de aceitação encontrados na bibliografia não são ótimos para este tipo de aplicação.

5.2.1.3 Análises das Métricas de Limiar e DTW

A Figura 5-18 apresenta a distribuição dos resultados das métricas Dvurecenska (VM), DTW e a derivada de DTW (DTW') da Planta Piloto de Neutralização de pH. A distribuição dos resultados de VM e DTW foi a esperada, bem balanceada e espalhada. O comportamento da métrica DTW' foi distinto, neste caso os valores estão concentrados em uma faixa bem pequena, mas o número de valores repetidos para comportamentos distintos dos modelos não é grande, portanto assume-se que a alta entropia não seja um problema para este conjunto de dados. A Tabela 5.12 apresenta algumas estatísticas deste conjunto de dados que apoiam estas conclusões.

Figura 5-18- Distribuição dos resultados das métricas VM, DTW e DTW' da Planta Piloto de Neutralização de pH.

Fonte: AUTOR.

Tabela 5.12- Estatísticas da distribuição dos resultados das métricas VM, DTW e DTW'.

	VM		DTW		DTW'	
	H	pH	H	pH	H	pH
Média	19.73	37.13	$1.69e^{+30}$	$6.1e^{+268}$	$2.39e^{+27}$	$4.25e^{+265}$
Desvio padrão	33.29	44.04	$8.53e^{+31}$	Inf	$1.22e^{+29}$	Inf
Mediana	0	0.8684	6.296	0.0585	$7.7e^{-05}$	0
Limite superior	90.8	99.21	$4.76e^{+33}$	$1.914e^{+272}$	$6.78e^{+30}$	$1.33e^{+269}$
Limite inferior	0	0	$1.55e^{-05}$	$2.3e^{-07}$	$1.09e^{-08}$	0
Modelos rejeitos (%)	76.71	61.95	4.61	3.37	-	-
Modelos aceitos (%)	23.29	39.05	95.39	96.63	-	-

Fonte: AUTOR.

Para calcular os limiares das malhas de pH e nível ($e_{th}=0,1561$ e $e_{th}=0,6145$), além de utilizar os valores dos dados de validação também foram empregados os dados de calibração dos instrumentos de medição apresentados no Capítulo 4. A Figura 5-19 descreve o comportamento dos limiares para as malhas de controle de nível (a) e pH (b). Pode-se observar como o ruído influencia os limiares.

Figura 5-19- Comportamento dos limiares das malhas de controle de nível (a) e pH (b).

Fonte: AUTOR.

As Figura 5-20 e Figura 5-21 exibem as respostas dos modelos números 1446, 1467, 19 e 1948. Os limiares obtidos permitem que dentro deles as respostas possam ter dinâmicas bastante diferentes da dinâmica da planta. Isto pode gerar falsas interpretações da métrica, porque nesse caso mesmo com bons resultados, o modelo pode não descrever a dinâmica certa. Os dados da Tabela 5.13 confirmam esta afirmação.

Figura 5-20- Respostas dos modelos números 1446 e 1467 da malha de pH.

Fonte: AUTOR.

Figura 5-21- Comportamento dos modelos números 19 e 1948 da malha de pH.

Fonte: AUTOR.

Tabela 5.13- Resultados das métricas VM, DTW e DTW' dos modelos números 1446, 1467, 19 e 1948 da malha de pH.

Model No.	VM(%)	e_{th}	DTW	DTW'	Ganho / (e_{ss})	$t_r(s)$	$t_s(s)$	$M_p(\%)$
1446	91.83	0.1561	0.0048	$1.14e^{-27}$	0.990 / (0.0679)	64	555	4.93
1467	87.48	0.1561	0.0040	0	0.991 / (0.0635)	31	404	48.18
19	93.25	0.1561	$5.43e^{-5}$	0	0.998 / (0.0151)	312	517	0
1498	87.11	0.1561	$8.60e^{-6}$	$2.92e^{-25}$	0.997 / (0.0198)	31	420	7.98

Fonte: AUTOR.

Os resultados das métricas do tipo DTW são interessantes. Esta métrica confirma sua sensibilidade para discriminar outros fatores da dinâmica, além do ganho. Pode-se observar na Figura 5-21 dois modelos com ganho similar, mas o valor da DTW é significativamente distinto. No caso da derivada da DTW o ruído faz que seja muito mais complicado interpretar os resultados. Os resultados desta métrica mostram que se consegue uma menor distância com modelos mais oscilatórios.

Planta de Clarke

A Figura 5-22 apresenta a distribuição dos resultados da métrica de Dvurecenska (VM), DTW e DTW'. Neste caso, VM mostra uma boa distribuição e uma porcentagem de acertos na faixa esperada. Este fato pode ser confirmado com as informações vistas na Tabela 5.14. O contrário ocorre com as métricas DTW e DTW', seus valores estão todos concentrados em uma faixa bem pequena, o que pode significar uma alta entropia. Este comportamento da DTW pode ser porque ela lida com distâncias euclidianas e o fato de trabalhar assim com um degrau de amplitude 1 vai condicionar bastante os resultados da DTW.

Figura 5-22- Distribuição dos resultados das métricas VM, DTW e DTW' na planta de Clarke.

Fonte: AUTOR.

Tabela 5.14- Estatística dos resultados das métricas VM, DTW e DTW' na planta de Clarke.

	VM (%)	DTW	DTW'
Média	54.31	$5.54e^{+14}$	$5.54e^{+14}$
Desvio padrão	27.09	$3.94e^{+16}$	$3.94e^{+16}$
Mediana	69.77	$7.07e^{-05}$	$5.38e^{-05}$
Limite superior	76.81	$2.80e^{+18}$	$2.80e^{+18}$
Limite inferior	0	$1.2e^{-35}$	0
Modelos rejeitos (%)	38.55	9.95	-
Modelos aceitos (%)	61.45	90.05	-

Fonte: AUTOR.

Na Figura 5-23 mostra-se o comportamento dos limiares de Dvurenceska para a planta de Clarke. Neste caso, o valor obtido foi $e_{th}=0,274$. A planta tem um ruído muito menor do que a planta de pH, assim ante mudanças menores há uma maior sensibilidade.

Figura 5-23- Comportamento dos limiares para a planta de Clarke.

Fonte: AUTOR.

A Figura 5-24 mostra uma comparação entre os modelos números 885, 2443, 4669 e 4720 e seus resultados de validação são vistos na Tabela 5.15. Os resultados de VM são similares às conclusões da análise visual, onde o comportamento oscilatório dos modelos números 4669 e 4720 não condizem com a resposta de um sistema estável ideal. Os resultados de DTW indicam uma boa sensibilidade ante as mudanças das respostas.

Figura 5-24- Resposta dos modelos números 885, 2443, 4669 e 4720 da planta de Clarke.

Fonte: AUTOR.

Tabela 5.15- Resultados da validação com as métricas VM, DTW e DTW' dos modelos números 885, 2443, 4669 e 4720 da planta de Clarke.

Model No.	VM(%)	e_{th}	DTW	DTW'	Ganho / (e_{ss})	$t_r(s)$	$t_s(s)$	$M_p(\%)$
885	55.94	0.274	0.0034	0.0009	0.979 / (0.0208)	25	51	0
2443	74.9	0.274	$2.87e^{-05}$	$1.05e^{-06}$	0.990 / (0.0099)	6	29	16.34
4669	61.17	0.274	0.1014	0.0076	0.989 / (0.0105)	11	217	29.76
4720	55.19	0.274	$3.37e^{-23}$	$5.78e^{-23}$	0.987 / (0.0126)	10	480	25.37

Fonte: AUTOR.

As observações acerca desta seção são:

- A métrica VM é altamente influenciada pelo ganho do modelo. Mas nos casos onde as plantas têm pouco ruído e os instrumentos estejam bem calibrados, valores maiores do que 65% indicam uma dinâmica confiável do modelo.
- As duas métricas têm uma sensibilidade maior ante *outliers* e ruído no conjunto de validação do que as métricas restantes vistas.
- DTW e DTW' avaliam que uma resposta senoidal é mais similar ao comportamento do ruído branco que uma reta com o valor médio desse ruído. Assim, valores muito pequenos têm alta probabilidade de indicar que o modelo tem *overfit*. Isto implica que DTW e DTW' têm maior chance de indicar como bom um modelo com *overfit*.

- DTW e DTW' dão maior importância a outros elementos da dinâmica além do ganho, mas a interpretação de seus resultados é mais complexa.

5.2.1.4 Análises dos Critérios de Seleção de Modelos

A Figura 5-25 exibe a distribuição dos resultados para as duas plantas dos distintos métodos de seleção de modelos usados neste trabalho. Os resultados destes métodos, exceto o FPE na malha de pH, mostram uma boa distribuição, como visto na Tabela 5.16.

Figura 5-25- Distribuição dos métodos de critérios de seleção de modelos das duas plantas: (a) planta de pH e (b) planta de Clarke.

Fonte: AUTOR.

Tabela 5.16- Estatísticas dos KPIs dos experimentos número 3 das duas plantas.

	AIC			BIC			FPE		
	Clarke	H	pH	Clarke	H	pH	Clarke	H	pH
Média	-6.060	-7.43	-11.57	-16036	-100910	-191911	0.04	1.81	0.03
Desvio padrão	7.71	13.85	13.44	38540	304672	295685	0.42	6.46	0.05
Mediana	-4.48	-0.55	-6.62	-8068	50349	-83115	0.01	0.57	0
Limite superior	3.36	5.58	-0.68	31053	185265	47486	28.75	265.16	0.5
Limite inferior	-73.36	-59.64	-59.05	-3.5e ⁻⁵	-1.3e ⁺⁶	-1.2e ⁺⁶	0	0	0

Fonte: AUTOR.

A interpretação dos resultados desta métrica é comparativa. Por isso, falar de um valor para aceitar ou rejeitar um modelo não é possível. Neste caso, o modelo que tenha

o menor valor é aquele que conseguiu perder menos informação na etapa de identificação. Esta interpretação é muito mais fácil de fazer quando se lida com dados sem ruído.

O fato dos resultados destas métricas estarem condicionados pelo experimento de identificação dificulta muito a sua interpretação pelos algoritmos de IA, porque o menor valor da identificação só garante que o modelo é uma boa representação daquele experimento de identificação específico. Como solução e tendo em conta o conhecimento prévio sobre os experimentos de identificação, os valores serão normalizados e penalizados dependendo da SNR dos experimentos de identificação. Esta solução, aplicável neste trabalho, é muito mais difícil de aplicar na indústria. Outra solução pode ser dar um valor qualitativo que descreva a qualidade dos dados de identificação e empregar esta informação como uma outra entrada.

Na Figura 5-25 (a) mostra-se o comportamento de modelos obtidos no experimento número 1 da malha de nível (vide o APÊNDICE B) e (b) apresenta-se o comportamento dos modelos obtidos no experimento número 3 da malha de nível. Os resultados, normalizados entre -2 e 2 e sem penalização estão na Tabela 5.17. Os resultados para o experimento número 1 da malha de nível têm uma coerência com o que idealmente seria um sistema estável. Mas é importante destacar que o pior valor possível não é um sistema instável, ao contrário, o pior modelo segundo estes métodos seria um modelo de primeira ordem estável. Contrário a este comportamento, os resultados dos experimentos número 3 são mais difíceis de interpretar. Por exemplo, os modelos números 270 e 1686 têm comportamentos bem similares, mas os valores das métricas dizem outra coisa. Além disso, os modelos que neste caso as métricas determinam que são os que menos informação perdem na identificação, têm um comportamento não ideal.

Figura 5-26- Resposta dos modelos números 3, 9, 19, 25, 270, 269, 1692 e 1686 da malha de nível.

Fonte: AUTOR.

Tabela 5.17- Resultados dos critérios de seleção dos modelos números 3, 9, 19, 25, 270, 269, 1692 e 1686 da malha nível.

Método	Model No.							
	3	9	19	25	270	269	1692	1686
AIC	2	1.9998	-1.9666	-1.9858	-2	-1.9163	0.263	0.2313
BIC	2	1.9999	-1.9670	-1.9862	-1.5778	-1.4881	0.3955	0.2686
FPE	2	1.9917	-2	-2	-2	-1.9198	0.2446	0.2129

Fonte: AUTOR.

5.2.1.5 Análise dos KPI na etapa de simulação

Na etapa de treinamento dos algoritmos, os dados obtidos acerca do desempenho das malhas de controle das duas plantas foram todos coletados de simulações, indicando que eles foram obtidos em condições ideais, portanto os dados dos KPI são o produto de simulações livres de ruído. O objetivo de usar simulações livres de ruído é conformar os dados de saída com menor incerteza e a menor entropia possível. Mas isso não quer dizer que não foram feitas simulações com ruído em outras fases.

Na Figura 5-27 são apresentadas as distribuições dos KPI para as malhas de controle da planta de pH. Os resultados dos KPI das duas plantas apresentam uma distribuição não desejada. Os casos mais críticos são o índice de Harris, IAU e a variabilidade. Estas métricas têm uma grande maioria de seus dados em uma faixa bem reduzida. Isto pode se dever ao fato que o MPC reage de um jeito muito parecido, mesmo com grandes mudanças nos modelos, ou que estes KPI não sejam adequados para estas situações.

Figura 5-27- Distribuição dos KPIs da planta de pH.

Fonte: AUTOR.

A Figura 5-28 mostra respostas da malha de nível controlada pelo MPC usando distintos modelos e na Tabela 5.18 são vistos os resultados de seus KPI. Segundo o índice de Harris a melhor resposta seria o MPC número 1235, mas que é instável. Analisando os resultados da Tabela 5.18 e da Figura 5-28 pode-se dizer que o índice de Harris não

fornece informação sobre a estabilidade do sistema; além de que não oferece nenhuma informação distintiva acerca dos outros elementos importantes da resposta da planta, como o erro estacionário e o tempo de acomodação. Além disso, tem uma baixa sensibilidade pois em modo servo o desvio padrão da saída (σ_{tot}) é significativamente maior que o desvio padrão dos valores consecutivos (σ_{cap}).

Figura 5-28- Resposta da malha de controle de nível controlada com os MPC usando os modelos números 1296, 1235, 1903 e 1902.

Fonte: AUTOR.

Tabela 5.18- KPIs da malha de controle de nível controlada com os MPC usando os modelos números 1296, 1235, 1903 e 1902.

Modelo No.	Harris	r_{wi}	IAU	MESR	MRSR	VAR	e_{ss}	$t_r(s)$	$t_s(s)$	Mp(%)
1296	3.90	3827	966881	0.0248	$5.6e^{-4}$	0.0421	0.0446	2161	4958	62.73
1235	1.45	$5.1e^{+27}$	208	0.9716	$3.4e^{-15}$	$5.2e^{-11}$	70	112	-1723	12.89
1903	3.22	21224	966601	0.0011	$1.1e^{-05}$	0.0323	$6.0e^{-11}$	114	227	4.17
1902	3.21	22523	966600	0.0011	$1.1e^{-05}$	0.0323	$5.8e^{-12}$	124	245	3.89

Fonte: AUTOR.

Este comportamento de pouca sensibilidade entre as respostas ocorre também nos resultados da variabilidade e do IAU. No caso do IAU é ainda pior, porque se trata de valores grandes e neste caso a diferença entre os MPC números 1903 e 1902 é só de 1. O restante dos KPIs mostrou um comportamento mais perto do desejado; destacando os que apresentam a maior sensibilidade. Mas ele teve a mesma desvantagem do índice de Harris, não dando informação sobre a estabilidade da resposta e assim um valor pequeno não indica uma resposta estável. Este comportamento ocorre porque tanto uma diminuição considerável do erro entre o *setpoint* e saída como um aumento grande na derivada média da saída podem ocasionar uma diminuição no valor do Watchdog.

Na Tabela 5.18 também são colocados os valores da resposta transitória das malhas com seus sistemas de controle. O comportamento destas métricas foi melhor do que os KPI pois seus resultados permitem uma melhor classificação das respostas.

Como já foi citado, a distribuição dos dados de saída e entrada do algoritmo é muito importante, e um comportamento totalmente indesejável na hora de conformar as classes de saídas é ter dados de saída muitos próximos uns dos outros ou repetidos. Uma solução comum nestes casos é normalizar as saídas para obter dados com uma distribuição mais uniforme, que permitam classificar melhor as respostas. Outra opção seria usar outras medidas para classificar as respostas, neste caso a resposta temporal das malhas.

A Figura 5-29 exibe a distribuição destas características das malhas de controle nas plantas usadas. Olhando para a Tabela 5.19 e a Figura 5-29 pode-se concluir que estes aspectos, menos o erro de estado estacionário (e_{ss}), fornecem uma informação com menor entropia, o que permite classificar melhor as respostas das malhas de controle.

Figura 5-29- Distribuição da resposta transitória da Planta Piloto de Neutralização de pH (a) e planta de Clarke (b).

Fonte: AUTOR.

Tabela 5.19- Estatísticas das respostas transitórias das duas plantas.

	e_{ss}			t_r (s)			t_s (s) ¹¹			M_p (%)		
	Clarke	H	pH	Clarke	H	pH	Clarke	H	pH	Clarke	H	pH
Média	8.26e ⁺²²	3.65	0.051	214	135	299	273	656	511	1.92	969.41	4.27
Desvio padrão	4.18e ⁺²⁴	15.3	0.608	311	242	220	360	871	467	64.7	50797	117.84
Mediana	4.4e ⁻¹⁶	5e ⁻⁹	1.2e ⁻¹⁰	31	103	176	50	643	422	0.22	11.84	0.42
Limite superior	2.35e ⁺²⁶	70	7.5	900	5730	2429	1000	4958	4958	4588.9	2.8e ⁶	6.2e3
Limite inferior	0	0	0	2	0	2	29	-1833	-1176	0	0	0
1 quartil	0	3.3e ⁻¹¹	5.4e ⁻¹³	25	97	147	40	344	240	0	4.9	0.02
3 quartil	0.0016	2.6e ⁻⁶	1.4e ⁻⁷	322	114	475	525	876	679	0.77	21.28	2.14

Fonte: AUTOR.

5.2.2 Definição das classes

Para os algoritmos de ML que têm função de classificação, o elemento principal no conjunto de saídas é a classe, a qual é um atributo especial que descreve o fenômeno de interesse que vai ser aprendido. A ação de classificar o comportamento do sistema segundo uma classe determinada é chamada catalogar. Neste caso, as classes representam a qualidade do sistema de controle, a qual é influenciada por distintos fatores, que foram descritos pelos KPIs da Seção 3.2. Portanto, para agrupar estes fatores em classes que ofereçam uma perspectiva geral do fenômeno, foi empregada a Equação (5.18) que calcula (ID_{Norm}). Os valores ID_{Norm} para cada exemplo são calculados empregando uma média ponderada dos KPIs escolhidos.

$$ID_{Norm} = w_{t_s} * t_s + w_{MESR} * MESR + w_{e_{ss}} * e_{ss} + w_{r_{wi}} * r_{wi} + w_{M_p} * M_p \quad (5.18)$$

onde w_{t_s} , w_{MESR} , $w_{r_{wi}}$, w_{Tr} e w_{ESS} são pesos dos respectivos KPIs. Os pesos usados aqui também foram escolhidos de forma empírica, observando-se a característica da distribuição dos KPIs como média e desvio padrão. Além disso, foi analisada a influência destes KPIs no comportamento da malha de controle. Baseado nestes critérios escolheu-se: $w_{t_s} = 30\%$, $w_{MESR} = 20\%$, $w_{e_{ss}} = 2,5\%$, $w_{r_{wi}} = 22,5\%$ e $w_{M_p} = 25\%$. Neste caso, os KPIs com maior influência têm um peso maior. É importante destacar que os valores de t_s , $MESR$, e_{ss} , r_{wi} , M_p utilizados nesta equação são normalizados com respeito aos valores de referência apresentados nos apêndices respectivos para cada planta. Este procedimento permite ter uma referência da distância entre o valor achado e de referência. O procedimento é descrito no Apêndice B.

A Tabela 5.20 apresenta como as classes foram delimitadas com respeito aos resultados da Equação (5.18) (ID_{Norm}).

¹¹ O tempo não pode ser negativo, se for isto indica que são sistemas instáveis, que não conseguem se estabilizar antes do segundo degrau.

Tabela 5.20- Declaração das classes.

		Planta/ Malha de controle		
		Clarke	Nível	pH
Nome da classe	Qualidade do controle	Limite de ID_{Norm}	Limite de ID_{Norm}	Limite de ID_{Norm}
A	Melhor (+)	[0 0.4]	[-1 0]	[0 0.5]
B		[0.4 0.6]	[0 0.5]	[0.5 1]
C		[0.6 0.8]	[0.5 1]	[1 1.5]
D		[0.8 1.5]	[1 2]	[1.5 2]
E		[1.5 8.5]	[2 3]	[2 2.5]
F		[8.5 100]	[3 4]	[2.5 3]
G	Pior (-)	[100 2000]	[4 6]	[3 4]
Z		[2000 Inf]	[6 Inf]	[4 Inf]

Fonte: AUTOR.

As três colunas finais da Tabela 5.20 apresentam os valores limites do índice ID_{Norm} para cada uma das variáveis controladas. Os limites foram escolhidos de forma empírica, mas com a ideia que os dados que compõem cada classe sejam o mais homogêneo possível. A premissa foi conformar classes onde o desvio padrão dos KPIs seja o menor possível, onde a média esteja centralizada, com uma quantidade considerável de exemplos e que forneça um conjunto balanceado (vide o Apêndice C).

Na classe 'Z' encontram-se também os modelos instáveis que não conseguem atender os critérios de estabilidade dos sistemas de controle ou têm um erro de estado estacionário maior que 6% do valor do *setpoint*. Como foi observado no item anterior, muitos dos KPIs não conseguem diferenciar entre sistema estáveis e instáveis e como se trabalha com uma grande quantidade de dados, não é possível fazer uma inspeção visual para cada um deles para determinar a sua estabilidade. Assim, para catalogar as respostas do sistema de controle na classe 'Z' não só é usada a Equação (5.18), mas também é usado um algoritmo sequencial que verifica se o sistema consegue se estabilizar em um valor e que este valor do tempo de acomodação tenha um erro na faixa de 6% da referência, que a resposta não oscile muito e, finalmente, que a variabilidade não aumente nos trechos finais da resposta, conforme descrito na Figura 5-30.

Figura 5-30- Fluxograma para determinar instabilidade.

Fonte: AUTOR.

5.3 Etapa de treinamento

A etapa de treinamento dos algoritmos de ML é um processo iterativo de prova e erro que visa achar os parâmetros e hiper-parâmetros que melhor solucionem a situação que deve ser aprendida. Os parâmetros são calculados de forma automática pelo algoritmo e sua obtenção depende exclusivamente do método de aprendizado usado. Os hiper-parâmetros¹² comumente são determinados pelo usuário do método, embora atualmente existam otimizadores que ajudam a calculá-los de modo automático. Ainda assim a escolha certa das entradas segue sendo manual e um processo imprescindível da etapa de treinamento. O otimizador bayesiano proposto em (MOČKUS, 1974) foi empregado para estimar os hiper-parâmetros, por isso o foco desta seção é descrever as relações entre algumas entradas e as classes para escolher as melhores entradas. Ainda assim, informação sobre todas as entradas pode ser encontrada no APÊNDICE D.

Na Figura 5-31 os resultados dos testes de S-W (utilizando o tempo de amostragem original) estão em uma faixa bem reduzida e muitos desses valores estão repetidos. Este comportamento similar nos outros testes de hipóteses é indesejável, confirma as observa-

¹² Os hiper-parâmetros mudam dependendo da pessoa que tenha programado os algoritmos. Neste trabalho utilizamos implementações do Matlab®.

ções do Item 5.2.1.1 e, por isso, não são usados na etapa de treinamento como entradas (vide o Apêndice D).

Figura 5-31- Relação entre os resultados de Shapiro-Wilk quando se usa o tempo de amostragem original e o ID_{Norm} da malha de pH.

Fonte: AUTOR.

A Figura 5-32 mostra a relação entre os valores de FIT e ID_{Norm} da malha de nível. Pode-se observar que modelos com FIT entre 0,6 e 0,7 têm uma maior tendência a ser instáveis do que estáveis e que há mais modelos com FIT negativo que geram um controle classe A, do que modelos com FIT entre 0,6 e 0,7. A quantidade de modelos com FIT negativo que gera um controle classes B e C é maior do que o resto dos modelos com FIT positivo. É evidente o comportamento díspar e não linear desta métrica com relação ao ID_{Norm} . Um comportamento similar também é notado na malha de pH (vide a Figura 5-33).

Figura 5-32- Relação entre o índice FIT e o ID_{Norm} da malha de nível.

Fonte: AUTOR.

Figura 5-33- Relação entre o índice FIT e o ID_{Norm} da malha de pH.

Fonte: AUTOR.

Na Figura 5-33 se nota também que a relação entre FIT e ID_{Norm} é não linear nem trivial, o que justifica o uso de IA para encontrar uma relação entre eles. Neste caso, também foram vistos comportamentos peculiares, mas distintos da malha de nível. Por exemplo, a menos dos modelos com FIT entre 0 e 0,1 o resto conseguiu gerar controles classe A. Mais uma vez os modelos com FIT negativo conseguiram uma quantidade maior de malhas de controle classes A e B que o resto. Além disso, também é interessante ver que mais modelos com FIT entre 0,8 e 0,9 controlaram a malha com uma qualidade C do que modelos com FIT entre 0,6 e 0,7.

A distribuição apresentada nas Figura 5-32 e Figura 5-33 têm em comum que a grande maioria dos modelos classe A encontram-se em uma faixa bem delimitada. Ainda quando esta faixa é mais restrita que os critérios de aceitação do Capítulo 3, esta nova faixa garante uma alta probabilidade que o controle seja bom. Este comportamento é repetido para o resto das métricas MAME em todas as plantas, portanto estas métricas são consideradas entradas de interesse (vide o APÊNDICE D).

A relação entre os valores de Dvurecenska e o ID_{Norm} (Figura 5-34) do modelo na malha de pH é similar ao comportamento visto na Figura 5-32. Destaca-se que na faixa entre 80-90% o número de modelos que produz um comportamento classe C é maior do que o esperado. Isto é provocado em boa medida pelo erro de calibração do medidor de pH.

Figura 5-34- Relação entre a métrica de Dvurenceska e o ID_{Norm} da malha de pH.

Fonte: AUTOR.

As métricas baseadas em critérios de seleção de modelos tiveram um bom comportamento, mostrando uma boa dispersão e em faixas determinadas podem ser discriminadas com clareza algumas classes, especialmente a classe A. A Figura 5-35 apresenta a relação entre AIC e ID_{Norm} da planta de Clarke.

Figura 5-35- Relação entre AIC e ID_{Norm} da planta de Clarke.

Fonte: AUTOR.

As figuras anteriores confirmam a dificuldade de empregar os métodos clássicos de validação para estas aplicações. Mas uma característica positiva que é repetida em quase todas é o fato de que aqueles modelos que geram uma boa performance estão

concentrados em uma faixa de valores bem clara. Também fica evidenciada a necessidade de usar várias entradas, principalmente para predizer as classes distintas de 'A' e 'Z'.

Os resultados da etapa de treinamento são exibidos de distintas formas. A Tabela 5.21 mostra as entradas escolhidas, os hiper-parâmetros estimados e a acurácia obtida.

Tabela 5.21- Características dos algoritmos de IA treinados.

Algoritmo IA	Malha de controle	Entradas escolhidas	Hiper-parâmetros	Acurácia
DT	Clarke	1.Willmott 2.Russell Fase 3.Dvurenceska 4.Anova 5.AIC 6.Tempo de Subida 7. Tempo de Estabelecimento 8.Sobressinal	Critério de Split: Gini index Número máximo de Split: 370	84.1 %
	Nível	1.TIC 2.Willmott 3.Russell Fase 4.Dvurenceska 5.Tempo de Subida 6.Tempo de Estabelecimento 7.Sobressinal	Critério de Split: Regra deTwoing Número máximo de Split: 348	87.9 %
	pH	1.TIC 2.Willmott 3.Russell Fase 4.Dvurenceska 5.Tempo de Subida 6.Tempo de Estabelecimento 7.Sobressinal	Critério de Split: Cross Entropy Número máximo de Split: 247	89.9 %
ANN	Clarke	1.TIC 2.Russell Fase 3.Dvurenceska 4.Sobressinal	Número de neurónios: 45	73.7 %
	Nível	1.TIC 2.Willmott 3.Russell Fase 4.Dvurenceska 5.Tempo de Subida 6.Tempo de Estabelecimento 7.Sobressinal	Número de neurónios: 97	70.1%
	pH	1.TIC 2.Russell Fase 3.Tempo de Subida	Número de neurónios: 98	71.9 %
SVM	Clarke	1.TIC 2.Russell Fase 3.Dvurenceska 4.Sobressinal	Kernel: Gaussiano Box constraint: 109,071	81.9 %
	Nível	1.TIC 2.Russell Fase 3.Tempo de Subida 4. Sobressinal	Kernel: Gaussiano Box constraint: 999.296	91.5 %
	pH	1.Willmott 2.Russell Fase 3.Tempo de Subida	Kernel: Gaussiano Box constraint: 693.125	91.0 %

Fonte: AUTOR.

As estruturas das árvores de decisão (DT) e das redes neurais (ANN) podem ser apresentadas de forma gráfica. No caso das DT encontram-se no Apêndice I e das ANN é mostrada na Figura 5-36.

Figura 5-36- Arquitetura da rede neural artificial usada na planta de Clarke.

Fonte: AUTOR.

Destaca-se que neste trabalho foi usada uma arquitetura multicamadas composta por uma camada oculta e uma camada de saída, o único elemento que varia é a quantidade de neurônios na camada oculta. Esta arquitetura é bem popular na classificação de problemas deste tipo. Além disso, a Figura 5-37 mostra uma relação da importância das distintas entradas no processo de treinamento das árvores de decisão, que é calculada somando o risco de mudança quando é efetuada uma divisão binária em cada uma das entradas.

Figura 5-37- Importância estimada das entradas das árvores de decisão para cada uma das malhas de controle.

Fonte: AUTOR.

Nesta dissertação, os algoritmos não foram treinados todos ao mesmo tempo. Usando o princípio da parcimônia, começou-se pelo mais simples de treinar, o DT. Observando os resultados deste e a importância das entradas apresentadas na Figura 5-37, decidiu-se que na hora de treinar os SVMs, deve-se realizar uma outra etapa de tratamento dos dados, a qual consta de duas ações: (i) truncar as entradas entre os valores do primeiro e do terceiro quartis da variável, (ii) normalizar esses valores entre -100 e 100.

5.4 Etapa de validação

Para validar o resultado desses algoritmos, várias métricas são usadas, quase todas suportadas pela Matriz de Confusão (Figura 5-38).

		Valor Real	
		1	0
Valor predito	1	Verdadeiros Positivos (TP)	Falso Positivo (FP)
	0	Falso Negativo (FN)	Verdadeiros Negativos (TN)

Figura 5-38- Matriz de Confusão.

Fonte: (LÓPEZ *et al.*, 2013).

Esta é uma matriz que resume os resultados preditos pelo classificador e os compara com os valores reais do conjunto de validação e que é comumente usada para avaliar o desempenho destes algoritmos. Associadas a essa matriz há várias métricas, três das quais são:

- Taxa de Verdadeiros Positivos (TP): $TPR = \frac{TP}{TP+FN}$ é a porcentagem de resultados positivos classificados corretamente.
- Taxa de Verdadeiros Negativos (TN): $TNR = \frac{TN}{TN+FP}$ é a porcentagem de resultados negativos classificados corretamente.
- Valor positivo predito ou precisão: $PPV = \frac{TP}{TP+FP}$ é a probabilidade que a predição positiva seja certa.

Outra métrica usada é a acurácia (Acc), dada pela Equação (5.19). Esta métrica oferece uma ideia da probabilidade de fazer uma predição certa pelos algoritmos. Mas esta informação vai depender, em parte, da distribuição do conjunto de dados e do limiar para definir as classes positivas e negativas. Portanto, se a probabilidade de observar uma classe em um conjunto de exemplos é muito maior do que observar outra, então altos

níveis de acurácia não indicam necessariamente um bom algoritmo (BRADLEY, 1997; LÓPEZ *et al.*, 2013).

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \quad (5.19)$$

A probabilidade de prever de modo correto um valor positivo aleatório pode ser estimada usando a área debaixo da curva (AUC) das características operacionais do Receptor (ROC). AUC-ROC tem uma alta concordância com a acurácia, mas com a vantagem de não depender da distribuição das classes. Esta métrica é calculada pelas Equações (5.20) até (5.24).

$$AUC = \sum_i \left\{ (1 - \beta_i \cdot \Delta\alpha) + \frac{1}{2} [\Delta(1 - \beta) \cdot \Delta\alpha] \right\} \quad (5.20)$$

$$\Delta(1 - \beta) = (1 - \beta_i) - (1 - \beta_{i-1}) \quad (5.21)$$

$$\Delta\alpha = \alpha_i - \alpha_{i-1} \quad (5.22)$$

$$(1 - \beta) = \frac{TP}{TP + FN} \quad (5.23)$$

$$\alpha = \frac{FP}{FP + TN} \quad (5.24)$$

Dividiu-se o conjunto de dados em três grupos: (i) treinamento, (ii) validação e (iii) teste. Os exemplos de cada um dos grupos são escolhidos aleatoriamente. O grupo de treinamento contém 70% do total. O grupo de validação contém 25% dos elementos, com os quais é feita a validação dos algoritmos treinados, facilitando avaliar o seu desempenho. O resto dos modelos é usado para testes, tanto na planta real como no simulador.

Os resultados de AUC da Tabela 5.22 descrevem o comportamento dos algoritmos com uma grande confiabilidade para prever as distintas classes. De modo geral, as classes 'A' e 'Z' apresentam os resultados mais confiáveis. Isto é muito importante, pois estas classes "extremas" são as mais críticas, em teoria se deseja ter performances de classe 'A' e evitar as de classe 'Z'.

Tabela 5.22- Resultados da AUC das classes dos algoritmos de IA treinados.

Algoritmo de IA	Malha de Controle	AUC da Classe							
		'A'	'B'	'C'	'D'	'E'	'F'	'G'	'Z'
DT	Clarke	0.94	0.89	0.95	0.94	0.91	0.94	0.93	0.98
DT	Nível	1	0.94	0.95	0.97	0.95	0.92	0.83	0.97
DT	pH	0.96	0.96	0.96	0.96	0.95	0.91	0.96	0.95
SVM	Clarke	0.97	0.93	0.96	0.96	0.93	0.97	0.96	0.97
SVM	Nível	1	0.99	0.99	0.97	0.99	0.97	0.99	1
SVM	pH	0.98	0.99	0.98	0.99	0.96	0.97	0.99	0.97
ANN	Clarke	0.97	0.93	0.96	0.90	0.91	0.96	0.96	0.99
ANN	Nível	0.99	0.97	0.95	0.91	0.92	0.91	0.96	0.99
ANN	pH	0.97	0.91	0.92	0.94	0.94	0.98	0.98	0.98

Fonte: AUTOR.

A Matriz de Confusão da Máquina de Vetores de Suporte (SVM) obtida na planta de Clarke apresenta resultados piores do que as da DT. Ainda que os valores da Tabela 5.22 sejam bons, os resultados da Figura 5-41 contrastam com estes. A principal limitante desta etapa de validação são os valores de TPR na faixa de 60%, valores muito baixos que não permitem afirmar que as predições nessas classes são confiáveis. Como aspectos positivos destaca-se, mais uma vez, que a distribuição dos erros de classificação é similar com o DT. Portanto, na maioria das vezes, errar significa pequenas diferenças entre os KPIs. Outro aspecto positivo é que essas classes com baixo TPR têm bom PPV.

SVM da planta Clarke

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	241	14	8	5	7	0	0	0	87.64%
	B	25	116	12	6	3	0	0	0	71.60%
	C	3	4	245	25	5	0	0	0	86.88%
	D	5	4	10	206	12	1	0	1	86.19%
	E	4	6	8	20	107	1	4	6	68.59%
	F	0	0	0	1	1	125	15	11	81.70%
	G	0	0	0	0	0	28	98	21	66.67%
	Z	0	0	0	0	0	4	9	174	93.05%
PPV		86.69%	80.56%	86.57%	78.33%	79.26%	78.62%	77.78%	81.69%	81.95%

Figura 5-41- Matrix de Confusão correspondente com o SVM da planta de Clarke.

Fonte: AUTOR.

A DT conseguiu prever com uma maior acurácia o comportamento dos sistemas de controle da malha de nível. A Figura 5-42 comprova que para todas as classes, com exceção da 'F' e 'G', o algoritmo obteve altos valores de TPR e PPV. O comportamento distinto destas classes pode ser devido a terem uma menor quantidade de exemplos. Ainda assim, a distribuição dos erros é muito similar com a DT da planta de Clarke, portanto estes erros de classificação não significam grandes diferenças nos KPIs das respostas.

DT da malha de Nível

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	124	0	0	1	0	0	0	0	99.20%
	B	0	39	3	0	0	0	0	0	92.86%
	C	0	5	57	3	1	2	0	0	83.82%
	D	0	0	3	86	5	0	0	0	91.49%
	E	0	1	1	13	163	9	0	0	87.17%
	F	0	0	0	8	11	73	4	0	76.04%
	G	1	0	0	0	0	13	40	10	62.50%
	Z	1	0	0	0	1	1	10	154	92.22%
PPV		98.41%	86.67%	89.06%	77.48%	90.06%	74.49%	74.07%	93.90%	87.31%

Figura 5-42- Matrix de Confusão correspondente com a DT da malha de nível.

Fonte: AUTOR.

Mais uma vez, a ANN da malha de nível teve um desempenho pobre comparado com o DT desta malha. Neste caso, acentuam-se os resultados observados na planta de Clarke, que as classes ‘A’ e ‘Z’ são preditas com uma alta confiabilidade enquanto as restantes não. Estes resultados são apresentados na Figura 5-42.

ANN da malha de Nível

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	91	7	0	0	0	0	0	0	92.86%
	B	12	20	0	0	0	0	0	0	62.50%
	C	10	10	17	2	14	0	0	0	32.08%
	D	0	0	16	42	31	2	0	2	45.16%
	E	0	0	1	13	112	14	4	0	77.78%
	F	0	0	0	2	50	20	8	1	24.69%
	G	0	0	1	0	3	2	45	9	75.00%
	Z	1	0	0	0	0	0	5	169	96.57%
PPV		79.82%	54.05%	48.57%	71.19%	53.33%	52.63%	72.58%	93.37%	70.11%

Figura 5-42- Matrix de Confusão correspondente com a ANN da malha de nível.

Fonte: AUTOR.

Contrário ao comportamento observado na planta de Clarke, o SVM teve melhores resultados que a DT. Mesmo assim, os resultados apresentados na Figura 5-43 são muito similares com o resultado da DT, a melhoria do SVM está em sua capacidade de fazer uma melhor classificação das classes ‘F’ e ‘G’. Neste caso, o SVM demonstrou ser uma melhor solução no caso de classes desbalanceadas.

SVM da malha de Nível

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	129	3	1	0	0	0	0	1	96.27%
	B	2	38	5	0	0	0	0	0	84.44%
	C	1	2	66	1	0	3	0	0	90.41%
	D	0	0	1	82	11	5	0	2	81.19%
	E	0	0	0	4	193	2	1	0	96.50%
	F	0	0	1	0	13	80	4	3	79.21%
	G	0	0	0	0	2	1	59	7	85.51%
	Z	1	0	0	0	0	0	3	215	98.17%
PPV		96.99%	88.37%	89.19%	94.25%	88.13%	87.91%	88.06%	94.30%	91.51%

Figura 5-43- Matrix de Confusão correspondente com o SVM da malha de nível.

Fonte: AUTOR.

A DT que valida os modelos usados na malha de pH teve a maior acurácia entre todas as DTs anteriores. Os resultados deste algoritmo mostrados na Figura 5-44 confirmam uma alta confiabilidade na hora de prever todas as classes. Mas a distribuição dos erros não ficou tão similar com as das malhas anteriores. Neste caso as classes ‘Z’ e ‘A’ evidenciam uma maior chance de cometer “grandes erros” (quando o erro de classificação

não está entre classes adjacentes). Segundo a Figura 5-44 estes “grandes erros” na maioria dos casos será a incorreta classificação na classe ‘Z’. Portanto, do ponto de vista da aplicação prática, este tipo de erro não ocasiona problemas na operação da planta.

DT da malha de pH

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	152	6	2	2	0	0	1	3	91.57%
	B	3	108	4	0	0	0	0	2	92.31%
	C	1	3	134	4	0	0	0	1	93.71%
	D	1	1	4	123	2	1	0	1	92.48%
	E	0	1	0	3	63	10	0	0	81.82%
	F	0	0	0	1	3	49	6	1	81.67%
	G	0	0	0	0	0	1	88	6	92.63%
	Z	1	0	0	0	4	3	7	73	82.95%
PPV		96.20%	90.76%	93.06%	92.48%	87.50%	76.56%	86.27%	83.91%	89.87%

Figura 5-44- Matrix de Confusão correspondente com a DT da malha de pH.

Fonte: AUTOR.

A validação da ANN concorda com os resultados anteriores. A performance deste algoritmo foi pior, porque neste caso as classes ‘A’ e ‘Z’ não tiveram uma confiabilidade tão alta quanto a dos casos anteriores (vide a Figura 5-45).

ANN da malha de pH

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	128	0	17	0	2	0	1	1	85.91%
	B	11	31	37	13	0	0	0	0	33.70%
	C	4	1	101	14	0	0	0	1	83.47%
	D	0	9	17	79	2	0	0	0	73.83%
	E	0	2	0	16	32	5	0	1	57.14%
	F	0	0	0	0	12	39	8	0	66.10%
	G	0	0	0	0	5	1	61	7	82.43%
	Z	1	0	1	1	2	3	11	56	74.67%
PPV		88.89%	72.09%	58.38%	64.23%	58.18%	81.25%	75.31%	84.85%	71.90%

Figura 5-45- Matrix de Confusão correspondente com a ANN na malha de pH.

Fonte: AUTOR.

A performance do SVM da malha de pH está em concordância com os resultados evidenciados na malha de nível. A acurácia é muito similar nas duas malhas. Ambos os algoritmos apresentam altos níveis de confiabilidade na hora de validar os distintos modelos (vide a Figura 5-46). Neste caso, destaca-se como elemento negativo a aparições dos “grandes erros”, sendo este fato mais crítico nas classes ‘A’ e ‘B’.

SVM da malha de pH

		Classe Predita								TPR
		A	B	C	D	E	F	G	Z	
Classe Real	A	166	3	2	0	0	0	0	6	93.79%
	B	2	119	5	0	0	0	0	0	94.44%
	C	1	6	134	10	1	0	0	0	88.16%
	D	0	3	8	131	1	0	0	0	91.61%
	E	0	1	3	8	63	5	1	0	77.78%
	F	0	1	0	0	3	60	0	0	93.75%
	G	0	0	1	0	1	4	94	2	92.16%
	Z	2	1	1	0	0	0	2	86	93.48%
PPV		97.08%	88.81%	87.01%	87.92%	91.30%	86.96%	96.91%	91.49%	91.04%

Figura 5-46- Matrix de Confusão correspondente com o SVM da malha de pH.

Fonte: AUTOR.

As Figura 5-47 e Figura 5-49 apresentam as respostas ao degrau de alguns dos modelos usados na planta de Clarke (vide a Tabela 5.25), assim como a resposta da malha de controle quando estes modelos são usados pelo controlador. Baseado nas informações observadas nestas figuras e na Tabela 5.23, pode-se evidenciar a capacidade dos algoritmos para prever os distintos comportamentos da malha de controle segundo o modelo. A Tabela 5.23 contém os resultados do FIT, TIC e VM para estes modelos assim como os resultados preditos pelos três algoritmos de ML treinados nesta planta. Fica evidente que mesmo tendo resultados aceitos pelo critério de validação, os modelos não produzem um bom desempenho do controlador MPC. Mais uma vez, pode-se comprovar que os erros de classificação na maioria dos casos estão entre classes adjacentes e como pode ser observado na Figura 5-50, as diferenças na análise visual entre as respostas da classe 'A', 'B' e 'C' são quase inexistentes. Como já foi dito anteriormente e mostrado no Apêndice C, estas classes discriminam diferenças médias entre 7 e 10 segundos de tempo de acomodação (vide a Tabela 5.24). Isto demonstra a precisão nas predições dos algoritmos, principalmente no caso do SVM.

Tabela 5.23- Comparação entre os resultados de validação e as classes preditas dos modelos números 885, 2443, 4720, 4489, 1636 e 4469 da planta de Clarke.

Modelo No.	Resultado de Validação			Classe Predita			Classe observada
	FIT	TIC	VM (%)	DT	ANN	SVM	
885	0.7972	0.1026	55.94	Z	G	Z	Z
2443	0.9296	0.0354	74.90	B	B	B	B
4720	0.8949	0.0528	55.19	Z	E	Z	Z
4489	0.9024	0.0507	49.19	B	A	A	A
1636	0.9232	0.0386	73.32	C	C	B	B
4469	0.7379	0.1501	21.48	C	C	C	C

Fonte: AUTOR.

Figura 5-47- Resposta ao degrau dos modelos números 4489, 1636 e 4469 identificados na planta de Clarke.

Fonte: AUTOR.

Figura 5-48- Comportamento da malha de controle da planta de Clarke quando o MPC utiliza os modelos números 885, 2443 e 4720.

Fonte: AUTOR.

Figura 5-49- Resposta ao degrau dos modelos números 4489, 1636 e 4469 identificados na planta de Clarke.

Fonte: AUTOR.

Figura 5-50- Comportamento da malha de controle da planta de Clarke quando o MPC utiliza os modelos números 4489, 1636 e 4469.

Fonte: AUTOR.

Tabela 5.24- Comparação entre os resultados dos KPIs quando são empregados os modelos números 885, 2443, 4720, 4489, 1636 e 4469 da planta de Clarke.

Modelo No.	KPI da malha de controle da planta Clarke				
	$t_s(s)$	MESR	ess	r_{wi}	Mp(%)
885	-	0.4409	2.66	1534716	0
2443	31.76	0.0028	$2.22e^{-16}$	256.59	1.65
4720	-	4.8929	0.51	16.83	0
4489	33.46	0.0027	$2.89e^{-14}$	252.67	1.09
1636	32.31	0.0029	0	245.38	1.57
4469	41.52	0.0032	0	362.02	0.5

Fonte: AUTOR.

Tabela 5.25- Polinômios dos modelos usados nas Figura 5-47 e Figura 5-49.

Modelo No. (Estrutura)	Polinômios da função de transferência	Modelo No. (Estrutura)	Polinômios da função de transferência
885 (ARMAX)	$A(z) = 1 - 2.578 z^{-1} + 2.29 z^{-2} - 0.7021 z^{-3}$ $B(z) = -0.02791 z^{-10} + 0.01409 z^{-11} + 0.01942 z^{-12} + 0.00409 z^{-13} + 7.161e^{-05} z^{-14}$ $C(z) = 1 + 0.9756 z^{-1} + 0.5274 z^{-2} + 0.3529 z^{-3} + 0.2562 z^{-4} + 0.3491 z^{-5} + 0.06699 z^{-6} + 0.7827 z^{-7} + 0.9114 z^{-8} + 0.3458 z^{-9} + 0.04423 z^{-10}$	4489 (ARMAX)	$A(z) = 1 - 2.952 z^{-1} + 3.148 z^{-2} - 1.354 z^{-3} + 0.1635 z^{-4} - 0.01459 z^{-5} + 0.0007201 z^{-6} - 0.00265 z^{-7} + 0.05716 z^{-8} - 0.0722 z^{-9} + 0.02914 z^{-10}$ $B(z) = -0.001275 z^{-1} + 0.01593 z^{-2} + 0.02329 z^{-3} - 0.02839 z^{-4} - 0.009649 z^{-5} + 0.001203 z^{-6} - 0.004847 z^{-7} + 0.006152 z^{-8} - 0.001598 z^{-9} + 0.00184 z^{-10}$ $C(z) = 1 - 2.169 z^{-1} + 1.545 z^{-2} - 0.3636 z^{-3}$
2443 (ARMAX)	$A(z) = 1 - 1.669 z^{-1} + 0.7425 z^{-2}$ $B(z) = -0.04885 z^{-1} + 0.1219 z^{-2}$ $C(z) = 1 - 1.051 z^{-1} + 0.2701 z^{-2} + 0.01785 z^{-3} - 0.02654 z^{-4} + 0.008613 z^{-5} - 0.006509 z^{-6} + 0.02535 z^{-7} - 0.02804 z^{-8} - 0.008055 z^{-9} + 0.03944 z^{-10}$	1636 (ARMAX)	$A(z) = 1 - 1.471 z^{-1} + 0.4557 z^{-2} + 0.1147 z^{-3}$ $B(z) = 0.1539 z^{-1} - 0.1937 z^{-2} - 0.07872 z^{-3} + 0.2528 z^{-4} - 0.08933 z^{-5} - 0.005378 z^{-6} + 0.3568 z^{-7} - 0.5846 z^{-8} + 0.3885 z^{-9} - 0.1022 z^{-10}$ $C(z) = 1 - 0.6129 z^{-1}$
4720 (OE)	$B(z) = 0.05462 z^{-1} - 0.1254 z^{-2} - 0.03513 z^{-3} + 0.4091 z^{-4} - 0.4717 z^{-5} + 0.1479 z^{-6} + 0.4263 z^{-7} - 1.143 z^{-8} + 1.17 z^{-9} - 0.4311 z^{-10}$ $F(z) = 1 - 2.893 z^{-1} + 2.767 z^{-2} - 0.605 z^{-3} - 0.4237 z^{-4} + 0.1569 z^{-5}$	4469 (ARMAX)	$A(z) = 1 - 1.915 z^{-1} + 1.032 z^{-2} - 0.03532 z^{-3} - 0.002524 z^{-4} + 0.01346 z^{-5} - 0.08024 z^{-6} + 0.05387 z^{-7} - 0.02679 z^{-8} + 0.02108 z^{-9} - 0.01736 z^{-10}$ $B(z) = 0.137 z^{-5} - 0.1046 z^{-6}$ $C(z) = 1 - 1.043 z^{-1} + 0.1641 z^{-2} + 0.03078 z^{-3} + 0.01381 z^{-4} + 0.03793 z^{-5}$

Fonte: AUTOR.

6 PERFORMANCE DOS ALGORITMOS DE IA NA PLANTA DE pH

Para emitir um critério sobre a performance dos algoritmos de IA na Planta Piloto de Neutralização de pH, comparam-se as classes preditas pelos algoritmos de IA com as classes obtidas na planta. Para calcular os valores de ID_{Norm} e assim determinar as classes obtidas, é repetido o procedimento explicado na Subseção 5.2.2. Mas, neste caso, os valores de referência para normalizar são outros (vide o Apêndice B). Isto é porque há certas diferenças entre o simulador e a planta real, mesmo quando o simulador é uma boa representação da planta. Neste capítulo vão ser comparados onze modelos para cada uma das malhas de controle da planta (vide o Apêndice J). Os resultados de algumas métricas de validação destes modelos, assim como os KPIs gerados ao usar esses modelos no MPC são mostrados na Tabela 6.1. Sobre esta tabela destaca-se o seguinte:

- Os KPIs de cada um dos experimentos de avaliação das malhas de controle são calculados com um MPC com os mesmos parâmetros de sintonia para cada malha.
- As métricas de validação analisadas foram escolhidas com base na importância que tem como entradas os algoritmos de IA. O índice FIT é incluso pela alta aplicação desta métrica na área de Identificação de Sistemas.
- Por conta do ruído da planta e amplitude do degrau, o critério para determinar t_s foi de 8% e 5% para as malhas de nível e pH, respectivamente.
- A coluna que contém as classes preditas é o resultado das predições do algoritmo de IA com os melhores resultados na etapa de validação. Neste caso são os SVM.

A Figura 6-1 contém a resposta ao degrau dos modelos números 19, 1930, 3117 e 3148, os quais são usados para descrever a malha de nível. Entre os modelos 3117 e 3148 e a resposta da planta podem ser observadas diferenças na dinâmica. A mais visível é a diferença entre os ganhos, mas como pode se ver na Tabela 6.1 também existem diferenças entre outras características dinâmicas da resposta. Como fruto destas diferenças, os resultados da maioria das métricas propostas de validação indicam estes modelos como inaceitáveis. Destaca-se o caso do FIT, no qual o resultado não está nem perto da região de aceitação. O resto dos modelos mostrados na Figura 6-1 foram aceitos por todas as métricas de validação.

Tabela 6.1- Resumo dos resultados dos modelos testados na Planta Piloto de Neutralização de pH.

Malha de Controle	Modelo No.	Resultados da Validação							KPIs da Malha de Controle						Classe	
		FIT	TIC	d'	P _r	VM(%)	t _r (s)	t _s (s)	M _p (%)	t _s (s)	MESR	e _{ss}	τ_{wi}	M _p (%)	Predita	Real
Nível	19	0.8673	0.0018	0.9043	0.0011	76.19	108	312	13.24	435	0.0054	0.01	1488.9	44.90	'A'	'A'
	1930	0.8552	0.0019	0.8866	0.0011	76.42	109	308	12.85	419	0.0052	0.007	1489.6	38.58	'A'	'A'
	3117	-0.0545	0.0138	-0.231	0.0011	0	95	417	10.86	414	0.0059	0.05	1673.2	41.08	'A'	'A'
	3148	-0.3563	0.0177	-0.4039	0.0011	0	119	412	11.55	425	0.0061	0.01	1859.2	41.34	'A'	'A'
	47	-0.7837	0.0243	-0.5468	0.0014	0	115	562	22.69	507	0.0058	0.002	1454.5	30.69	'B'	'B'
	158	-3.0597	0.057	-0.8016	0.0012	0	107	458	15.91	435	0.0055	0.01	1460.9	33.62	'B'	'A'
	1932	0.6687	0.0044	0.6059	0.0011	61.19	89	306	12.1	566	0.0062	0.016	1919.7	47.50	'B'	'B'
	962	-2.5171	0.0446	-0.771	0.0014	0	216	305	0	751	0.0063	0.028	1680	35.73	'E'	'D'
	1131	-13.44	0.2374	-0.9443	0.0032	0	210	572	0.35	1618	0.009	0.006	1309	13.58	'G'	'G'
	1142	-27.30	0.6021	-0.9716	0.0049	0	675	1036	0	Inf.	0.015	Inf.	2185	32.41	'Z'	'Z'
	1145	-20.67	0.4046	-0.9629	0.0038	0	524	778	0	1947	0.011	0.110	1591.2	26.24	'Z'	'Z'
pH	2350	0.4337	0.0084	0.4207	0.0014	88.23	468	869	0	159	0.0033	0.0015	227.52	7.8	'A'	'A'
	2347	0.5477	0.0067	0.5489	0.0014	90.84	485	925	0	252	0.0030	0.0018	151.33	24.6	'A'	'A'
	2344	0.5213	0.0071	0.5191	0.0014	90.17	471	916	0	213	0.0045	0.012	192.96	63.4	'A'	'A'
	2395	0.2962	0.0105	0.2697	0.0013	77.00	447	805	0	302	0.0053	0.0053	134.14	83.4	'A'	'A'
	160	0.8629	0.002	0.923	0.0013	93.25	312	517	0	534	0.0044	0.0058	159.67	59.4	'A'	'C'
	2428	-13.9	0.1816	-0.9337	0.0296	0	3768	4466	0	548	0.0061	0.0061	195.70	42.4	'C'	'C'
	2443	-13.6	0.1786	-0.9325	0.0286	0	3739	4451	0	561	0.0044	0.0058	159.67	42.6	'C'	'C'
	11	0.1284	0.0131	0.1012	0.0019	10.8	287	394	0	557	0.0065	0.0053	188.62	87.8	'C'	'D'
	14	0.1189	0.0132	0.1006	0.0022	13.63	470	639	0	778	0.0058	0.0042	102.11	59.4	'E'	'E'
	464	-2.37	0.0523	-0.6916	0.0145	0	2276	3009	0	Inf.	0.0223	0.0216	205.94	146.6	'Z'	'Z'
	465	-1.76e ⁻⁷	1	-1	0.6216	0	3246	4977	0	Inf.	Inf.	Inf.	Inf.	-	'Z'	'Z'

Fonte: AUTOR.

Figura 6-1- Resposta ao degrau dos modelos números 3117, 3148, 19 e 1930 da malha de nível.

Fonte: AUTOR.

Ainda assim, o resultado de usar estes modelos para o controle foi satisfatório para todos os casos. Tanto na planta real como em seu simulador, os modelos conseguiram controlar com uma performance catalogada de 'A'. A Figura 6-2 mostra o comportamento da malha de controle de nível (abaixo) e o *setpoint* fornecido pelo MPC ao PID (acima). Nesta é evidenciado um comportamento similar para cada um dos modelos testados.

Figura 6-2- Comportamento da malha de nível quando o MPC emprega os modelos números 19, 1930, 3117 e 3148.

Fonte: AUTOR.

A inspeção visual da Figura 6-2 indica que as respostas da malha de controle são quase idênticas, em teoria deveriam estar catalogadas na mesma classe. No entanto, os

resultados da Tabela 6.1 mostram que o comportamento usando o modelo 3148 foi mal predito. A principal discrepância observada neste caso foi um aumento do índice Watchdog com respeito ao valor de referência. Este KPI é altamente influenciado pela variabilidade do sinal e, portanto, pelas perturbações da planta. Este erro de classificação não é crítico, mas é importante destacá-lo.

A Figura 6-4 contém a resposta ao degrau dos modelos números 47, 158 e 1932, que são usados para descrever a malha de nível. Neste caso, as diferenças na característica da resposta transitória são maiores para os modelos números 47 e 158. Em consequência, só o TIC define estes modelos como uma boa representação da planta, isto segundo o critério de aceitação proposto em (ROWLAND; HOLMES, 1978).

Figura 6-3- Resposta ao degrau dos modelos números 47, 158 e 1932 da malha de nível.

Fonte: AUTOR.

Mesmo com estas diferenças entre os três modelos, na etapa de simulação o uso deles provoca um controle catalogado de 'B' em cada um dos casos. O resultado de empregar os modelos da Figura 6-3 para controlar a malha de nível é apresentado na Figura 6-4. Neste caso, as predições dos algoritmos de IA não geraram uma classificação certa no caso do modelo 158, que teve um comportamento catalogado de 'A'. Este resultado pode ser produto do ruído e perturbações da planta. Isto é inferido pelo fato dos valores dos KPI serem muito similares, a maior diferença está no t_s e só de 64 s, e pelas similitudes entre os *setpoints* fornecidos pelo MPC ao PID.

Figura 6-4- Comportamento da malha de nível quando o MPC emprega os modelos números 47, 158 e 1932.

Fonte: AUTOR.

Finalmente, a resposta ao degrau dos modelos números 962, 1131, 1142 e 1145 da malha de nível é apresentada na Figura 6-5. Os comportamentos dos modelos mostrados aqui estão ainda mais afastados do comportamento da planta do que os modelos anteriormente mencionados. Em concordância com este fato, a maioria dos métodos de validação determinam que estes modelos não são aceitáveis.

Figura 6-5- Resposta ao degrau dos modelos números 962, 1131, 1142 e 1145 da malha de nível.

Fonte: AUTOR.

O resultado de empregar estes modelos para o controle está mais de acordo com a premissa de que modelos não aceitos pelas métricas de controle deveriam gerar um controle ruim da planta. Na etapa de simulação é predito que usando o modelo 962 gera

uma resposta classe ‘E’, usando o modelo 1131 produz uma resposta classe ‘G’ e os restantes casos gera uma resposta classe ‘Z’. O resultado de usar estes modelos é visto na Figura 6-6. Neste caso, os resultados preditos pelo algoritmo e a planta real concordam para todos os casos, excetuando o modelo 962. A maior diferença observada foi o tempo de acomodação, comparando este com o tempo médio achado nesta classe a diferença é significativa. Este erro de classificação poderia ser considerado como crítico, mas o fato de que a PPV desta classe é de 78% e que neste caso a performance foi melhor que o esperado são elementos positivos deste comportamento.

Figura 6-6- Comportamento da malha de nível quando o MPC emprega os modelos números 962, 1131, 1142 e 1145.

Fonte: AUTOR.

A Figura 6-7 contém a resposta ao degrau dos modelos números 2350, 2347, 2344 e 2395, os quais são utilizados para descrever a malha de pH. Todos estes modelos têm diferenças na dinâmica com respeito à planta. A mais visível é a diferença entre os ganhos, outras características da resposta transitória são apresentadas na Tabela 6.1. Destacar que esta diferença nos ganhos faz com que os modelos sejam rejeitados pela maioria dos métodos de validação, excetuando-se TIC e Dvurecenska.

Figura 6-7- Resposta ao degrau dos modelos números 2350, 2347, 2344 e 2395 da malha de pH.

Fonte: AUTOR.

O resultado de usar estes modelos no controle foi satisfatório para todos os casos. Tanto na planta real como no simulador, os modelos controlaram com um desempenho nível 'A'. A Figura 6-8 mostra o comportamento da malha de controle de nível e o *setpoint* fornecido pelo MPC ao PID. Em ambos os casos as respostas são quase idênticas. O ID_{Norm} medido da planta concorda com o valor predito pelos algoritmos (vide Tabela 6.1).

Figura 6-8- Comportamento da malha de pH quando o MPC emprega os modelos números 2350, 2347, 2344 e 2395.

Fonte: AUTOR.

A Figura 6-9 mostra a resposta ao degrau dos modelos números 160, 2428 e 2443, os quais são usados para descrever a malha de pH. Neste caso, as diferenças na carac-

terística da resposta transitória são maiores para os modelos números 2428 e 2443. Como consequência, a maior parte dos métodos de validação os definem como inaceitáveis.

Figura 6-9- Resposta ao degrau dos modelos números 160, 2428 e 2443 da malha de pH.
Fonte: AUTOR.

O resultado de empregar os modelos da Figura 6-9 para controlar a malha de pH é mostrado na Figura 6-10. Neste caso, para o modelo 160 foi predito um comportamento do sistema de controle de classe 'A', mas na realidade foi classe 'C'. O *setpoint* imposto pelo MPC quando este modelo é usado tem uma dinâmica mais lenta do que utilizando os outros modelos, portanto este erro de classificação não é considerado produto das perturbações ou incertezas que podem ocorrer neste tipo de experimento, mas à incapacidade do algoritmo de prever.

Figura 6-10- Comportamento da malha de pH quando o MPC emprega os modelos números 160, 2428 e 2443.

Fonte: AUTOR.

Finalmente, as respostas ao degrau dos modelos números 11, 14, 464 e 465 são vistas na Figura 6-11. Elas estão menos afastadas da planta real do que os modelos anteriormente vistos. Embora com melhores resultados na validação que os modelos da Figura 6-10, o desempenho da malha de controle é pior e às vezes instáveis (vide a Figura 6-12). Os resultados concordam tanto na fase de simulação quanto nos resultados reais na planta.

Figura 6-11- Resposta ao degrau dos modelos números 11, 14, 464 e 465 da malha de pH.

Fonte: AUTOR.

Figura 6-12- Comportamento da malha de pH quando o MPC emprega os modelos números 11, 14, 464 e 465.

Fonte: AUTOR.

7 CONCLUSÕES E TRABALHOS FUTUROS

O emprego da IA para validar de forma automática e confiável o modelo a ser utilizado por um MPC mostrou ser uma boa solução. Em três situações distintas (mudando o MPC e o processo) estes métodos tiveram um bom desempenho. Neste caso, a capacidade de predição destes algoritmos é similar ou maior que em trabalhos anteriores como (MORALES ALVARADO, 2017) e (SARIN et al., 2010). Para conseguir estes resultados, o trabalho foi focado em quatro áreas fundamentais: (i) Validação de Modelos, (ii) Avaliação das Malhas de Controle, (iii) Aprendizagem Automática e (iv) Controle Preditivo baseado em Modelo. Sobre cada um destes temas este trabalho ofereceu conclusões e sugestões que o autor acha interessante destacar neste capítulo.

Sobre a validação de modelos:

- Os testes de hipóteses não fornecem informação determinante para decidir se o modelo produz uma boa resposta do MPC. Estes métodos são muito influenciados pelo número de amostras, conjunto de dados com mais de 300 exemplos tem uma tendência muito alta de rejeitar a hipótese nula. Como consequência, validar/invalidar um modelo baseado em rejeitar a hipóteses nula não é recomendado. É importante destacar que quando se comparam malhas estáveis em faixas lineares com modelos lineares e estáveis, a distribuição estatística dos resíduos vai ser determinada principalmente pela distribuição do ruído da malha.
- As métricas MAMEs usadas neste trabalho são fortemente influenciadas pelas diferenças de ganho entre o modelo e a planta quando se trabalha em malhas fechadas e estáveis. A capacidade de mostrar informação sobre o comportamento médio dos erros de predição é boa, mas esta informação não é suficiente para validar os modelos para estas aplicações.
- O critério para aceitar/rejeitar modelos usando o índice TIC apresentado em (ROWLAND; HOLMES, 1978) não é apropriado para esta aplicação. O valor de 0,4 significa que o resíduo representa mais do 40% da soma das respostas medidas e preditas. Neste trabalho concluiu-se que para este tipo de aplicação, o resíduo deveria representar menos que 6% da soma de ambas as respostas ($TIC < 0,06$). As probabilidades de que o modelo com esses valores de TIC produza um controle não desejável são muito baixas, como se observa no Apêndice D.

- O uso da métrica de Dvurecenska mostrou-se uma solução simples e adequada. Ela é muito influenciada pela qualidade dos dados de validação e pelo erro de calibração dos instrumentos. Nas condições deste trabalho, os resultados foram muito bons (especialmente na planta de Clarke e na malha de nível da Planta Piloto de Neutralização de pH). Quando os dados de validação têm pouco ruído e os instrumentos estão bem calibrados, esta métrica não só é capaz de distinguir diferenças de ganho, mas também é sensível a outras mudanças na dinâmica (tempo de acomodação, tempo de subida, etc.) que as MAMEs não distinguem claramente.
- As métricas DTW e DTW' mostraram resultados contrastante. Contrariamente com as métricas MAMEs têm uma sensibilidade maior as mudanças entre as dinâmicas da planta e o modelo. Neste caso comportamentos oscilatórios e diferenças das pendentes são detectados. Porém o fato de trabalhar com dados de validação ruidosos, valores pequenos podem significar que o modelo é uma boa representação do ruído e não do processo. Delimitar quando os resultados descrevem um modelo certo o um modelo com *overfitting* é muito complexo pois a métrica não cumpre com o princípio de desigualdade triangular.

Sobre a avaliação das malhas de controle:

- Nem o índice de Harris nem o *Watchdog* conseguem fornecer informação coerente sobre a estabilidade do sistema de controle. Este fato faz com que resultados de valores menores destas métricas não signifiquem melhorias no controle e sim na instabilidade. Isto não permite emitir um critério sobre a qualidade do controle só com estas informações. O índice de Harris também é afetado pelas mudanças no *setpoint*, portanto quando o controle está em modo servo os resultados são menos confiáveis ainda. Isto faz com que o índice de Harris não tenha sido considerado para calcular ID_{Norm} . Não se recomenda o uso de Harris nestas aplicações.
- Os métodos baseados em estatística fornecem informações mais fáceis de interpretar. Neste caso, pode-se notar mudanças dependendo do tempo de acomodação e do erro de estado estacionário. Ainda assim, estas não têm muita variabilidade e para um mesmo processo estável não apresentam variações muito grandes.
- Os valores de IAU não conseguiram os resultados esperados. O fato de ter os valores muito concentrados em uma faixa muito pequena impossibilitou seu uso para conformar o índice qualidade do controle ID_{Norm} . Se mostra como um bom elemento para comparar, mas não para catalogar.

- Ainda que não muito usadas na indústria, as características da resposta transitória provaram ser muito úteis para classificar as respostas do sistema de controle. Estes elementos facilitam muito a interpretação do que está acontecendo na malha de controle em modo servo, sem ter que ficar olhando diretamente a resposta.

Sobre a Aprendizagem Automática

- As entradas propostas para prever o comportamento do sistema de controle não tiveram o mesmo desempenho. Neste trabalho, usar todos os métodos de validação propostos não produziu o melhor resultado de predição. Os melhores resultados foram obtidos com um conjunto de dados de entradas com limites estabelecidos e simetricamente distribuídos. Neste caso foram: TIC, Willmott, Dvurecenska, Fase de Russell.
- Ainda que o FIT seja uma métrica de referência em muitos trabalhos de Identificação de Sistemas, para as aplicações estudadas neste trabalho não é recomendado. O fato de não ter limite produz uma baixa performance nos algoritmos de predição. Para estes trabalhos aconselha-se substituir o FIT pelo índice de Willmott que tem limite entre -1 e 1 e são calculados de forma muito similar, com resultados aproximadamente iguais quando a média do erro é menor que o desvio padrão da resposta da planta.
- A distribuição dos resultados de validação apresenta muitas classes superpostas. Mas isto não foi um impedimento para que os SVM obtiveram o melhor desempenho. Para consegui-lo, um fator fundamental foi tratamento dos dados antes da fase de treinamento.
- O SVM teve uma performance muito boa com valores de acurácia de 81,9%, 91,5% e 91% para as malhas da planta de Clarke, nível e pH, respectivamente. Este algoritmo mostrou sua capacidade para classificar classes com comportamento altamente não linear e ter uma alta confiabilidade de predição. Como elemento negativo destaca-se o trabalho da fase de treinamento deste algoritmo.
- A DT é sugerida como a melhor solução ao problema que serve como motivação deste trabalho. De modo geral, os resultados da validação são satisfatórios, com valores de acurácia de 84,1%, 87,9% e 89,9% para as malhas da planta de Clarke, nível e pH, respectivamente. Mesmo sendo superado pelo SVM nas malhas de nível e pH, a diferença é muito pequena. Além disso, a distribuição dos erros nas DTs favorece-as mais neste tipo de aplicação. Outro fator a considerar é a facilidade

de que tem para ser treinada e, de todos os algoritmos, é o mais simples e de menor custo computacional. Finalmente, se nos baseamos no princípio da parcimônia, não existe uma melhor solução.

- As ANN tiveram resultados mais fracos devido às limitações para lidar com classes superpostas. Tendo em conta sua comprovada capacidade de reconhecimento de padrões, é possível que os resultados negativos podem ser devidos às arquiteturas da rede utilizada ou à fase de processamento dos dados antes do treinamento. Ainda assim, os resultados não são totalmente negativos, pelo fato de que as ANN foram capazes de prever com precisão as classes 'A' e 'Z'. Provavelmente, com uma abordagem distinta, os resultados destes métodos poderiam melhorar.
- Os resultados observados na etapa de validação confirmam que estas previsões têm um alto nível de confiabilidade. Mas é importante destacar que os resultados do sistema de controle na planta são influenciados pelo ruído e pela diferença nos tempos de subida e de acomodação entre a planta simulada de Clarke e a planta real. Um simulador mais preciso poderia produzir uma melhor performance dos algoritmos na planta real.

Sobre o Controle Preditivo baseado em Modelo

- Mesmo quando um modelo que seja uma boa representação da planta devesse produzir um comportamento desejado na malha de controle, isto não exclui que um modelo com uma dinâmica diferente não possa criar o mesmo comportamento. Ficou demonstrado que prever o comportamento do sistema de controle não é um problema trivial, porque modelos muito diferentes podem gerar respostas iguais.
- Os métodos MAMEs, os quais são os mais populares nesta área, não fornecem informação necessária para definir a aptidão do modelo para gerar uma boa resposta do MPC. Pois como foi visto neste trabalho, diferenças de ganho não é o principal fator que influencia o comportamento do sistema de controle.
- O critério para validar os modelos para estas aplicações tem que ser criado usando distintas informações (métodos de validação), mas este vai mudar dependendo do controlador preditivo e do processo controlado.

7.1 Propostas de trabalhos futuros

As sugestões de trabalhos futuros nesta linha de pesquisa são dadas a seguir:

- Aprofundar em como automatizar e melhorar este processo de pré-processamento dos dados para treinar os algoritmos de IA.
- Utilizar outros métodos de IA, como *Deep Learning* ou *Random Forest* para tentar validar modelos para estas aplicações.
- Usar métricas que possam descrever outros comportamentos da resposta temporal, como por exemplo um comportamento oscilatório.
- Utilizar visão computacional na validação dos modelos.

REFERÊNCIAS BIBLIOGRÁFICAS

- ABB. **Functional Description PID01-Closed Loop Controller**. [s.l: s.n.]. Disponível em: <<https://library.e.abb.com/public/311634e891e7ee35c12579b8002e5f84/3BTG81179-2-3032-Functional-Description-PID01.pdf>>. Acesso em: 3 set. 2019.
- ADEBIYI, A. A.; ADEWUMI, A. O.; AYO, C. K. Comparison of ARIMA and Artificial Neural Networks Models for Stock Price Prediction. **Journal of Applied Mathematics**, v. 2014, p. 1–7, 5 mar. 2014.
- AGUIRRE, L. A. **Introdução à Identificação de Sistemas – Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais**. [s.l.] Editora UFMG, 2000.
- AKAIKE, H. Fitting Autoregressive Models for prediction. **Annals of the Institute of Statistical Mathematics**, v. 21, n. 1, p. 243–247, 1969.
- AKAIKE, H. **Information theory and an extension of the maximum likelihood principle**. 2nd International Symposium on Information Theory, Tsahkadsor, Armenia, USSR. **Anais...**1973
- AKAIKE, H. Information Theory and an Extension of the Maximum Likelihood Principle. In: [s.l.] Springer, New York, NY, 1998. p. 199–213.
- ALMUALLIM, H.; KANEDA, S.; AKIBA, Y. Development and applications of Decision Trees. In: **Expert Systems**. [s.l: s.n.]. p. 53–77.
- ALPAYDIN, E. **Introduction to Machine Learning**. 2nd. ed. Cambridge: [s.n.].
- ANDERSON, T. W. **Anderson-Darling Tests of Goodness-of-Fit**. [s.l: s.n.]. Disponível em: <http://statweb.stanford.edu/~ckirby/ted/papers/2010_Anderson-Darling.pdf>. Acesso em: 23 jun. 2019.
- AURIA, L.; MORO, R. A. Support Vector Machines (SVM) as a Technique for Solvency Analysis. **DIW Berlin German Institute for Economic Research Discussion Paper Series**, p. 18, 2008.
- BEMPORAD, A.; RICKER, N. L.; MORARI, M. **Model Predictive Control Toolbox™ User's Guide**. Natick, MA: [s.n.].
- BEMPORAD, A.; RICKER, N. L.; MORARI, M. **Model Predictive Control Toolbox™ Getting Started Guide**. Natick, MA: [s.n.].
- BEN SAIDA, A. **Shapiro-Wilk and Shapiro-Francia normal tests**Mathworks, , 2014. Disponível em: <<https://www.mathworks.com/matlabcentral/fileexchange/13964-shapiro-wilk-and-shapiro-francia-normality-tests>>
- BERENSON, M. L.; LEVINE, D. M.; KREHBIEL, T. C. Wilcoxon Signed Ranks Test: Nonparametric Analysis for Two Related Populations. In: **Basic Business Statistics: Concepts and Applications**. Twelfth ed. [s.l.] Prentice Hall, 2012. p. 890.
- BLIEMEL, F.; MACKAY, D. B. Theil's Forecast Accuracy Coefficient: A Clarification. **Journal of Marketing Research**, v. 10, p. 444–447, 1973.
- BOUM, A. T.; LATIFI, M. A.; CORRIOU, J. P. Observer based and Quadratic dynamic matrix control of a fluid catalytic cracking unit: A comparison study. **International Journal of Computer Applications**, v. 80, n. 3, p. 8, 2013.

- BOUTABA, R. et al. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. **Journal of Internet Services and Applications**, v. 9, n. 1, p. 104, 1 maio 2018.
- BOZDOGAN, H. THE GENERAL THEORY AND ITS ANALYTICAL EXTENSIONS. **PSYCHOMETRIKA**, v. 52, n. 3, p. 345–370, 1987.
- BRADLEY, A. P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. **Pattern Recognition**, v. 30, n. 7, p. 1145–1159, 1 jul. 1997.
- BRIEMAN, L. et al. **Classification and Regression Trees**. [s.l.] Taylor & Francis, 1984.
- CAMACHO, E. F.; BORDONS, C. (CARLOS). **Model Predictive Control**. 2nd. ed. London: Springer London, 2007.
- CLARKE, D. W.; MOHTADI, C.; TUFFS, P. S. Generalized Predictive Control—Part II Extensions and interpretations. **Automatica**, v. 23, n. 2, p. 149–160, 1 mar. 1987.
- COLLOBERT, R.; BENGIO, S. SVMTool: Support Vector Machines for Large-Scale Regression Problems. **Journal of Machine Learning Research**, v. 1, p. 143–160, 2001.
- CORREA DE GODOY, R. J. **Sintonia ótima de controladores**. [s.l.] Universidade de São Paulo, 2012.
- CREVIER, D. **AI: The Tumultuous Search for Artificial Intelligence**. New York, NY: Basic Books, Inc, 1993.
- CURSOR. **Identificação da Planta Piloto de Neutralização de pH DO LCPI / USP**. São Paulo: [s.n.].
- CUTLER, C. R.; RAMAKER, B. L. **Dynamic matrix control: A computer control algorithm**. Joint Automatic Control Conference. **Anais...** San Francisco, CA, USA: 1979
- DARBY, M. L.; NIKOLAOU, M. MPC: Current practice and challenges. **Control Engineering Practice**, v. 20, p. 238–342, 2012.
- DAS, K.; BEHERA, R. N. A Survey on Machine Learning: Concept, Algorithms and Applications. **International Journal of Innovative Research in Computer and Communication Engineering**, v. 5, n. 2, p. 1301–1310, 2017.
- DEGROOT, M. H.; SCHERVISH, M. J. **Probability and Statistics**. 4th. ed. [s.l.] Addison-Wesley, 2012.
- DESVAGES, H.; STABILITO, M. **Arquitetura de Redes**. São Paulo: [s.n.].
- DEVROYE, L.; GYÖRFI, L.; LUGOSI, G. **A Probabilistic Theory of Pattern Recognition**. 31th. ed. New York: Springer, New York, NY, 1996.
- DOWNEY, B. et al. A system identification approach for developing model predictive controllers of antibody quality attributes in cell culture processes. **Biotechnology Progress**, v. 33, n. 6, p. 1647–1661, 2017.
- DVURECENSKA, K. et al. A probabilistic metric for the validation of computational models. **Royal Society Open Science**, v. 5, p. 14, 2018.
- FABOZZI, F. J. et al. Model Selection Criterion: AIC and BIC. In: **The Basics of Financial Econometrics: Tools, Concepts, and Asset Management Applications**. [s.l.] John

Wiley & Sons, 2014. p. 399–403.

FAVARO, J. **Controle preditivo aplicado à planta piloto de neutralização de pH**. São Paulo: Biblioteca Digital de Teses e Dissertações da Universidade de São Paulo, 2012.

FORBES, M. G. et al. **Model Predictive Control in Industry: Challenges and Opportunities**. 9th International Symposium on Advanced Control of Chemical Processes. **Anais...** British Columbia, CA: International Federation of Automatic Control, 2015

FRANCO-ARCEGA, A.; FLORES-FLORES, L. G.; GABBASOV, R. F. **Application of Decision Trees for Classifying Astronomical Objects**. 2013 12th Mexican International Conference on Artificial Intelligence. **Anais...** IEEE, nov. 2013 Disponível em: <<http://ieeexplore.ieee.org/document/6714666/>>. Acesso em: 14 jun. 2019

GARCIA, C. **Controle de processos industriais**. 2nd. ed. São Paulo, SP: Blucher, 2017.

GARCIA, C. E.; MORSHEDI, A. M. Quadratic Programming Solution of Dynamic Matrix Control (QDMC). **Chemical Engineering Communications**, v. 46, n. 1–3, p. 73–87, 20 jul. 1986.

GARCIA MARQUES, F. **Modelagem fenomenológica e controle de uma planta de piloto de neutralização de pH**. [s.l.] Universidade de São Paulo, 2015.

GEERS, T. L. An objective error measure for the comparison of calculated and measured transient response histories. **The Shock and Vibration Bulletin**, v. 54, p. 99–107, 1984.

GÓRECKI, T.; ŁUCZAK, M. Using derivatives in time series classification. **Data Mining and Knowledge Discovery**, v. 26, n. 2, p. 310–331, mar. 2013.

GRUPPE, H. **Controlling Process KPIs A Guideline for Measuring Performance in Controlling Processes**. [s.l.: s.n.]. Disponível em: <<http://dnb.ddb.de>>. Acesso em: 31 jul. 2019.

H. RAY, M. Repeatability of Full-Scale Crash Tests and Criteria for Validating Simulation Results. **Transportation Research Record**, v. 1528, p. 155–160, 1996.

HARRIS, T. J. Assessment of Control Loop Performance. **The Canadian Journal of Chemical Engineer**, v. 67, p. 856–861, 1989.

HAYKIN, S. **Neural Networks and Learning Machines**. 3rd. ed. Hamilton, ON: Pearson Prentice Hall, 2009.

HONC, D. et al. Teaching and Practicing Model Predictive Control. **IFAC-PapersOnLine**, v. 49, n. 6, p. 34–39, 2016.

IBM. **CART Algorithms**. Disponível em: <https://www.ibm.com/support/knowledgecenter/en/SSLVMB_22.0.0/com.ibm.spss.statistics.algorithms/alg_tree-cart.htm>. Acesso em: 5 set. 2019.

JOE QIN, S. Control performance monitoring — a review and assessment. **Computers & Chemical Engineering**, v. 23, n. 2, p. 173–186, 26 jul. 2002.

KAT, C.-J.; ELS, P. S. Validation metric based on relative error. **Mathematical and Computer Modelling of Dynamical Systems**, v. 18, n. 5, p. 487–520, 2012.

- KLEIJEN, J. P. C. **Validation of models: statistical techniques and data availability.** WSC'99. 1999 Winter Simulation Conference Proceedings. "Simulation - A Bridge to the Future". **Anais...**1999
- LEOTTI TORMAN, V. B.; COSTER, R.; RIBOLDI, J. Normality of variables: Diagnosis methods and comparison of some nonparametric tests by simulation. **HCPA**, v. 32, p. 227–234, 2012.
- LIPTÁK, B. G. **Instrument Engineers's Handbook Process Control and Optimization.** Boca Raton, FL: Taylor & Francis Group, 2006.
- LIU, H. et al. Comparison of four Adaboost algorithm based artificial neural networks in wind speed predictions. **Energy Conversion and Management**, v. 92, p. 67–81, 1 mar. 2015.
- LIU, Y. et al. Toward a Better Understanding of Model Validation Metrics. **Journal of Mechanical Design**, v. 133, p. 13, 2011.
- LJUNG, L. **System identification : theory for the user.** New Jersey: Prentice Hall PTR, 1999.
- LOAGUE, K.; GREEN, R. E. Statistical and graphical methods for evaluating solute transport models: Overview and application. **Journal of Contaminant Hydrology**, v. 7, n. 1–2, p. 51–73, 1 jan. 1991.
- LÓPEZ, V. et al. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. **Information Sciences**, v. 250, p. 113–141, nov. 2013.
- LORENZEN, M.; CANNON, M.; ALLGÖWER, F. Robust MPC with recursive model update. **Automatica**, v. 103, p. 461–471, 2019.
- LUENGO, J.; GARCÍA, S.; HERRERA, F. A study on the use of statistical tests for experimentation with neural networks: Analysis of parametric test conditions and non-parametric tests. **Expert Systems with Applications**, v. 36, n. 4, p. 7798–7808, 1 maio 2009.
- LUMLEY, T.; SCOTT, A. AIC and BIC for modeling with complex survey data. **Journal of Survey Statistics and Methodology**, p. 18, 2015.
- MASSEY, F. J. The Kolmogorov-Smirnov Test for Goodness of Fit. **Journal of the American Statistical Association**, v. 46, n. 253, p. 68–78, 1951.
- MATHWORKS. **One-sample Kolmogorov-Smirnov test - MATLAB kstest.** Disponível em: <<https://www.mathworks.com/help/stats/kstest.html>>. Acesso em: 23 jun. 2019a.
- MATHWORKS. **Anderson-Darling test - MATLAB adtest.** Disponível em: <https://www.mathworks.com/help/stats/adtest.html?s_tid=doc_ta>. Acesso em: 23 jun. 2019b.
- MATHWORKS. **Box plot - MATLAB boxplot.pdf.** Disponível em: <<https://www.mathworks.com/help/stats/boxplot.html>>.
- MATHWORKS. **Akaike's Information Criterion for estimated model - MATLAB aic.** Disponível em: <<https://www.mathworks.com/help/ident/ref/aic.html>>. Acesso em: 5 set. 2019a.
- MATHWORKS. **Akaike's Final Prediction Error for estimated model - MATLAB fpe.**

Disponível em: <<https://www.mathworks.com/help/ident/ref/fpe.html>>. Acesso em: 5 set. 2019b.

MATHWORKS. **Box Plots - MATLAB & Simulink**. Disponível em: <<https://www.mathworks.com/help/stats/box-plots.html>>. Acesso em: 18 fev. 2019c.

MCCORDUCK, P. **Machines Who Think**. 2nd. ed. Natick, MA: A K Peters, Ltd, 2004.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, v. 5, n. 4, p. 115–133, dez. 1943.

MOČKUS, J. **On bayesian methods for seeking the extremum**. IFIP Technical Conference on Optimization Techniques. **Anais...**Novosibirsk: Springer Verlag, 1974

MONGIARDINI, M. et al. **Verification and validation of models used in computer simulations of roadside barrier crashes**. Proceedings of the 2013 Australasian Road Safety Research, Policing & Education Conference 28th. **Anais...**Brisbane, Queensland: 2013

MORALES ALVARADO, C. S. **Identificação e controle preditivo de uma planta-piloto de neutralização de pH**. [s.l.] Universidade de São Paulo, 2013.

MORALES ALVARADO, C. S. **Estudo e implementação de métodos de validação de modelos matemáticos aplicados no desenvolvimento de sistemas de controle de processos industriais**. [s.l.] Universidade de São Paulo, 2017.

MORALES ALVARADO, C. S.; GARCIA, C. Implementation of KPIs for Analyzing Control Loop Performance by using PI System of the OSIsoft Enterprise. **IEEE ASSP MAOAZINE**, v. 16, n. 1, p. 8, 2018.

NAGHILI, H. et al. Validation of drop plate technique for bacterial enumeration by parametric and nonparametric tests. **Veterinary Research Forum**, v. 4, p. 179–183, 5 ago. 2013.

NISBET, R.; ELDER, J. F. (JOHN F.; MINER, G. **Handbook of statistical analysis and data mining applications**. [s.l.] Academic Press/Elsevier, 2009.

OBERKAMPF, W. L.; BARONE, M. F. Measures of agreement between computation and experiment: Validation metrics. **Journal of Computational Physics**, v. 217, n. 1, p. 5–36, 1 set. 2006.

OGATA, K. **Ingeniería de control moderna**. 5a. ed. Madrid, España: Pearson Educación, 2010.

PACHECO DA SILVA, R. **Introdução à Identificação de Sistemas utilizando o System Identification Toolbox do MATLAB em conjunto com o Arduino para o Laboratório de Controle Linear**. [s.l.] Universidade Estadual Paulista, 2015.

PÉREZ SERRANO, M. A. **Estratégias de controle avançado para uma planta de tratamento de pH**. [s.l.] Universidade de São Paulo, 2019.

PÉREZ SERRANO, M. A. et al. **Sintonia e Avaliação de Controladores PID para uma Planta Piloto de Neutralização de pH**. XIV Conferência Brasileira de Dinâmica, Controle e Aplicações. **Anais...**São Carlos, SP: 2019

PILLONETTO, G.; CHIUSO, A.; NICOLAO, G. DE. Prediction error identification of linear systems : A nonparametric Gaussian regression approach. **Automatica**, v. 47, n. 2,

p. 291–305, 2011.

PORFÍRIO, C. R.; ALMEIDA NETO, E.; ODLOAK, D. Multi-model predictive control of an industrial C3/C4 splitter. **Control Engineering Practice**, v. 11, n. 7, p. 765–779, 2003.

RAHMAN, M. M.; GOVINDARAJULU, Z. A modification of the test of Shapiro and Wilk for normality. **Journal of Applied Statistics**, v. 24, n. 2, p. 219–236, abr. 1997.

RAY, M. H. Repeatability of Full-Scale Crash Tests and Criteria for Validating Simulation Results. **Transportation Research Record**, v. 1528, p. 155–160, [s.d.].

RAZALI.N.M; WAH.Y.B. Power comparisons of Shapiro-Wilk , Kolmogorov-Smirnov , Lilliefors and Anderson-Darling tests. **Journal of Statistical modeling and analytics**, v. 2, n. 1, p. 21–33, 2011.

REYES DREKE, V. D.; PÉREZ SERRANO, M. A.; GARCIA, C. **An approach to model validation for Model Predictive Control based on Dvurecenska's metric**. 14th International Conference on Automatic Control and Soft Computing. **Anais...**Bragança, Portugal: 2020

RHINEHART, R. R. **A watchdog for controller performance monitoring**. Proceedings of 1995 American Control Conference - ACC'95. **Anais...**Seattle, WSH: American Autom Control Council, 1995Disponível em: <<http://ieeexplore.ieee.org/document/531368/>>. Acesso em: 12 jul. 2019

RICE, R. C.; JYRINGI, R. R.; COOPER, D. J. **Performance Monitoring Fundamentals: Demystifying Performance Assessment Techniques**. [s.l: s.n.]. Disponível em: <[https://bin95.com/Performance Monitoring Fundamentals-2.pdf](https://bin95.com/Performance%20Monitoring%20Fundamentals-2.pdf)>. Acesso em: 6 set. 2019.

ROMANUS, E. E.; GIOVANINNI, R. V. **Estudo, implementação e testes de diferentes algoritmos de controle aplicados a uma planta de neutralização de pH**. [s.l.] Universidade de São Paulo, 2014.

ROUSSEEUW, P. J.; LEROY, A. M. **Robust Regression and Outlier Detection**. Hoboken, NJ, USA: John Wiley & Sons, Inc., 1987.

ROWLAND, J. R.; HOLMES, W. M. Simulation validation with sparse random data. **Computers & Electrical Engineering**, v. 5, n. 1, p. 37–49, 1 mar. 1978.

ROYSTON, P. Approximating the Shapiro-Wilk W-test for non-normality. **Statistics and Computing**, v. 2, p. 117–119, 1992.

RUEL, M. **Learn how to assess and improve control loop performance**. ISA2002. **Anais...**Chicago, IL: 2002Disponível em: <[https://www.idc-online.com/technical_references/pdfs/instrumentation/LEARN HOW TO ASSESS AND IMPROVE CONTROL LOOP PERFORMANCE.pdf](https://www.idc-online.com/technical_references/pdfs/instrumentation/LEARN_HOW_TO_ASSESS_AND_IMPROVE_CONTROL_LOOP_PERFORMANCE.pdf)>. Acesso em: 9 set. 2019

RUSSELL, D. M. **Error Measures for Comparing Transient Data: Part I: Development of a Comprehensive Error Measure**. The Proceedings of the 68th Shock and Vibration Symposium. **Anais...**1997

RUSSELL, S. J.; NORVIG, P.; EDWARDS, D. D. **Artificial Intelligence: A Modern Approach**. 3rd. ed. Upper Saddle River, New Jersey: Prentice Hall, 2010.

SAKOE, H.; CHIBA, S. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. **IEEE Transactions on Acoustics, Speech, and Signal Processing**,

v. 26, n. 1, p. 43–49, 1978.

- SALVADOR, S.; CHAN, P. **FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space**. Workshop on Mining Temporal and Sequential Data. **Anais...**2004Disponível em: <<https://pdfs.semanticscholar.org/05a2/0cde15e172fc82f32774dd0cf4fe5827cad2.pdf>>. Acesso em: 23 set. 2019
- SARIN, H. et al. **A Comprehensive Metric for Comparing Time Histories in Validation of Simulation Models With Emphasis on Vehicle Safety Applications**. ASME International, 24 jul. 2009
- SARIN, H. et al. Comparing Time Histories for Validation of Simulation Models: Error Measures and Metrics. **Journal of Dynamic Systems, Measurement, and Control**, v. 132, 2010.
- SCHWARZ, G. Estimating the Dimension of a Model. **The Annals of Statistics**, v. 6, n. 2, p. 461–464, mar. 1978.
- SHESKIN, D. J. **Parametric and non-parametric statistical procedures**. 2nd. ed. [s.l.] Chapman&Hall, 2000.
- SHIER, R. **The Wilcoxon signed rank sum test**. [s.l.: s.n.]. Disponível em: <<http://www.statstutor.ac.uk/resources/uploaded/wilcoxonsignedranktest.pdf>>. Acesso em: 24 jun. 2019.
- SHUNTA, J. P. **Achieving world class manufacturing through process control**. [s.l.] Prentice Hall PTR, 1995.
- SÖDERSTRÖM, T.; STOICA, P. **System Identification**. Upper Saddle River, NJ, USA: Prentice Hall, 1989.
- SONG, Y.-Y.; LU, Y. Decision tree methods: applications for classification and prediction. **Shanghai archives of psychiatry**, v. 27, n. 2, p. 130–135, 25 abr. 2015.
- SORNETTE, D. et al. Algorithm for model validation: Theory and applications. **Proceedings of the National Academy of Sciences**, v. 104, n. 16, p. 6562–6567, 17 abr. 2007.
- SPRAGUE, M. A.; GEERS, T. L. A spectral-element method for modelling cavitation in transient fluid-structure interaction. **INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING**, v. 60, p. 2467–2499, 2004.
- STAMP, M. A Survey of Machine Learning Algorithms and Their Application in Information Security. In: PARKINSON, S.; CRAMPTON, A.; HILL, R. (Eds.). **Guide to Vulnerability Analysis for Computer Networks and Systems: An Artificial Intelligence Approach**. 1st. ed. [s.l.] Springer International Publishing, 2018. p. 33–55.
- SUYKENS, J. A. K.; DE MOOR, B. L. R.; VANDEWALLW, J. P. L. **Artificial Neural Networks for Modelling and Control Non-Linear Systems**. [s.l.] SPRINGER-SCIENCE+BUSINESS MEDIA, B.V., 1996.
- THACKER, B. H. et al. **Concepts of Model Verification and Validation**. [s.l.: s.n.].
- The Shapiro-Wilk and related tests for normality**. . [s.l.: s.n.]. Disponível em: <<https://math.mit.edu/~rmd/465/shapiro.pdf>>. Acesso em: 23 jun. 2019.

- THEIL, H. **Applied Economic Forecasting**. Amsterdam: North Holland Publishing Company, 1966.
- TULLEKEN, H. J. A. F. Generalized binary noise test-signal concept for improved identification-experiment design. **Automatica**, v. 26, n. 1, p. 37–49, 1 jan. 1990.
- VAN WOENSEL, T.; VANDAELE, N. Empirical validation of a queueing approach to uninterrupted traffic flows. **Journal of Operations Research**, v. 4, p. 59–72, 2006.
- VOZÁK, D.; VESELÝ, V. Robust Model Predictive Controller Design. **IFAC Proceedings Volumes**, v. 47, n. 3, p. 7443–7448, 1 jan. 2014.
- WANG, L. **Support Vector Machines: Theory and Applications**. Berlin: Springer, 2005.
- WILLMOTT, C. J. Some Comments on the Evaluation of Model Performance. **Bulletin American Metrological Society**, v. 63, n. 11, p. 1309–1313, 1982.
- WILLMOTT, C. J.; ROBESON, S. M.; MATSUURA, K. A refined index of model performance. **International Journal of Climatology**, v. 32, n. 13, p. 2088–2094, 15 nov. 2012.
- XI, Y.-G.; LI, D.-W.; LIN, S. Model Predictive Control — Status and Challenges. **Acta Automatica Sinica**, v. 39, n. 3, p. 16, 2013.
- YAMAOKA, K.; NAKAGAWA, T.; UNO, T. Application of Akaike's information criterion (AIC) in the evaluation of linear pharmacokinetic equations. **Journal of Pharmacokinetics and Biopharmaceutics**, v. 6, n. 2, p. 165–175, 1978.
- YOKOGAWA. **PH200 2-Wire Type pH Meter General Specification** Tokyo, 1997.
- YOKOGAWA. **EJA 110A Differential Pressure Transmitter General Specifications** Tokyo, Japan, 2012.
- YUAN, H. et al. Neural network-based fault detection method for aileron actuator. **Applied Mathematical Modelling**, v. 39, n. 19, p. 5803–5815, 1 out. 2015.
- YUCHUN TANG et al. SVMs Modeling for Highly Imbalanced Classification. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, v. 39, n. 1, p. 281–288, fev. 2009.
- ZIEGLER, J. G.; NICHOLS, N. B. Optimum Settings for Automatic Controllers. **Transaction of the ASME**, v. 64, p. 759–768, 1942.

APÊNDICE A – Planta de Clarke

O *benchmark* de Clarke é uma planta proposta em (CLARKE; MOHTADI; TUFFS, 1987) que tem dois elementos principais: a "Planta" que simula o processo e o "Controlador", responsável por manter a saída do processo próxima à referência. A Figura A.1 mostra uma imagem geral do sistema. Nela, o bloco "Planta" é composto por uma função de transferência discreta que descreve a dinâmica do processo e também possui vários elementos geradores de ruído na saída (à esquerda da Figura A.2). O outro elemento do sistema é o bloco "Controlador" (à direita da Figura A.2), que descreve um controlador PI configurado com *anti-reset windup*, cujos parâmetros de sintonia são: ganho proporcional (K_c) igual a 1,75 e tempo integral (T_i) igual a 10 s/rep. O intervalo de amostragem (T) é de 1 segundo.

Figura A.1- Representação da Planta de Clarke usando o *Simulink*®.

Fonte: AUTOR.

A planta em malha aberta se comporta como mostrado na Figura A.3, representando um processo estável e superamortecido de ganho igual a 1, tempo de acomodação (t_s) igual a 21 s, segundo o critério de 5% apresentado em (OGATA, 2010).

Figura A.2- Conteúdo dos blocos principais da planta, “Planta” (a) e “Controlador” (b).

Fonte: AUTOR.

Figura A.3- Resposta da Planta de Clarke em malha aberta.

Fonte: AUTOR.

Quando está em malha fechada, a planta se comporta como um sistema subamortecido com ganho unitário, sobressinal (M_p) de 12,7%, t_s de 27 s e tempo de subida (t_r) de 10 s usando o critério de 0-100% apresentado em (OGATA, 2010).

Figura A.4- Resposta da Planta de Clarke em malha fechada com um degrau unitário no *setpoint*.

Fonte: AUTOR.

Para efetuar o Teste nº 1 foram usados os três sinais exibidos nas Figura A.5 e Figura A.6. No caso do teste com os dois primeiros sinais, não se incluiu ruído na saída da planta, mas no teste com o terceiro sinal sim. Para gerar o ruído foram usados os seguintes valores de variância no bloco “Perturbação”: $0,5e^{-7}$, $85e^{-7}$, $15e^{-6}$, $3e^{-5}$, $5e^{-5}$.

Figura A.5- Experimento de identificação em malha fechada usando vários degraus.

Fonte: AUTOR.

Destaca-se que no projeto do sinal GBN, considerou-se que as probabilidades de comutação (p_{sw}) escolhidas foram 0,005 e 0,125 para alta e baixa frequências, respectivamente.

Figura A.6- Experimento de identificação em malha fechada usando GBN lento.

Fonte: AUTOR.

Figura A.7- Experimento de identificação em malha fechada usando GBN rápido.

Fonte: AUTOR.

Depois de obtido um conjunto de dados como foi visto nas figuras anteriores, é feita a identificação dos modelos com o *Ident Toolbox* do Matlab®, sendo usados

modelos paramétricos com estruturas ARX, ARMAX e OE. Embora os MPC dos tipos DMC e QDMC tenham sido projetados com modelos do tipo resposta ao degrau (FSR) e resposta ao impulso (FIR), estes modelos têm limitações que as estruturas ARX, ARMAX e OE não têm, as quais podem representar sistemas instáveis, precisam de menos parâmetros para fazer uma boa estimação e é fácil obter um modelo do tipo FSR ou FIR a partir delas.

Para obter uma grande quantidade de modelos com distintos comportamentos foram identificados modelos com distintas ordens, mas usando os mesmos experimentos. A Tabela A.1 apresenta a relação das estruturas, as ordens e a quantidade dos modelos identificados com um conjunto de dados do Teste nº 1. A combinação entre as ordens dos polinômios determina a quantidade possível de modelos a serem identificados por cada estrutura.

Tabela A.1- Características dos modelos obtidos no Teste nº 1.

Estrutura	Ordens dos polinômios	Atraso	Quantidade de modelos
ARX	na=[1 3 2 5 10]; nb=[1 3 2 5 10]	nk=[1 5 10]	75
ARMAX	nf=[1 3 2 5 10]; nb=[1 3 2 5 10]; nc=[1 3 2 5 10]	nk=[1 5 10]	375
OE	nf=[1 3 2 5 10]; nb=[1 3 2 5 10]	nk=[1 5 10]	75

Fonte: AUTOR.

O resultado do Teste nº 1 é um conjunto de modelos a ser usado no Teste nº 2. Este contém 5250 modelos divididos em grupos de 750 do tipo ARX, 3750 do tipo ARMAX e 750 do tipo OE. Os grupos têm 30% dos modelos obtidos usando degrau, 40% usando GBN lento e 30% usando GBN rápido.

A Figura A.8 apresenta as respostas de alguns dos modelos identificados que são excitados por um terceiro sinal GBN que tem uma semente distinta dos GBN dos testes anteriores e um p_{sw} igual a 0,005. Nesta etapa da validação é adicionado na saída da planta um ruído “moderado” com variância $2e^{-6}$ para simular as plantas reais, mas na saída dos modelos tem-se só o sinal limpo.

Figura A.8- Predição infinitos passos à frente dos modelos identificados.

Fonte: AUTOR.

A Tabela A.3 mostra algumas das características de uma parte dos modelos obtidos na identificação. Neste caso, o FIT é calculado para predições 10 passos à frente e para infinitos passos à frente. A grande disparidade entre alguns valores dos índices FIT é devida ao fato de se utilizar sinais com altos níveis de ruído, fazendo que para um horizonte de predição menor, o modelo seja mais parecido com a variável afetada por ruído.

Tabela A.2- Informações sobre os modelos da Figura A.8.

No. do modelo	Estrutura	Ordem dos polinômios ¹³	Dados de identificação	Ruído da planta	FIT (k=10)	FIT (k=∞)
13	ARX	na=1; nb= 10; nk=1	Figura A.5	0	89.79%	90.98%
32	ARX	na=2; nb= 1; nk=5	Figura A.5	0	78.63%	76.46%
2178	ARMAX	na=10; nb= 1; nc=1, nk=10	Figura A.7	$85e^{-7}$	-129.33%	46.95%
4512	OE	na=1; nb= 5; nk=10	Figura A.5	0	12.95%	12.95%

Fonte: AUTOR.

Como a planta é simulada e sem restrições de controle ou de processo, isso torna o controlador DMC um ótimo algoritmo MPC. A Figura A.9 exibe a malha com o MPC.

¹³ O valor nk corresponde ao atraso puro do modelo identificado.

Figura A.9- Diagrama da planta controlada com um MPC.

Fonte: AUTOR.

Os parâmetros de sintonia do MPC são: intervalo de amostragem (T) de 1 segundo, horizonte de predição (p) de 10 segundos, horizonte de controle (m) de 2 segundos, valor nominal de MV e CV igual a 0, peso do incremento da variável manipulada (ϑ) de 2,22553; peso da saída (γ) de 0,44933.

A Equação (A.1) mostra o modelo de referência da planta, o qual descreve de forma quase perfeita o comportamento da planta em malha fechada. Porém, a resposta da planta controlada com um MPC que usa este modelo é considerada como a resposta de referência. Sabendo-se que o sistema em malha fechada com o controlador PI é estável e subamortecido com ganho unitário, um modelo paramétrico do processo é obtido usando o *Ident Toolbox* do Matlab®.

$$G(s) = \frac{(1 + 9,0854 * s)e^{-0.648*s}}{(1 + 2 * 0,36794 * 2,9655 * s + (2,9655 * s)^2)(1 + 11,4634 * s)} \quad (A.1)$$

A Figura A.10 mostra uma comparação das duas respostas e mostra também o resultado do controle da planta usando um controlador MPC com estes modelos. Nesta figura comprova-se a capacidade do MPC de atuar em condições ruidosas.

Figura A.10- Resposta da planta controlada com um MPC com modelo de referência e ruído.

Fonte: AUTOR.

Figura A. 11- Resposta da planta controlada com um MPC com modelo de referência sem ruído.

Fonte: AUTOR.

Os valores da resposta da planta mostrados na Figura A. 11 são os valores de referência usados para calcular o conjunto de saída, os quais são observados na Tabela A.3. O controle desta planta usando o MPC melhorou a performance, voltando-a mais rápida ao novo *setpoint* (diminui o t_s) e robusta, além de eliminar o sobressinal.

Tabela A.3- KPIs de referência da Planta de Clarke.

KPIs	Valor de referência x_{ref}	Valor de base x_{basis}
Índice de Harris (η_{Harris})	7,9840	-
Índice Watchdog (r_{wi})	191	191
IAU	2325.3198	-
Erro de estado estacionário (ess)	0,0024	0,01
Tempo de subida (t_r)	29 s	-
Tempo de estabelecimento (t_s)	30 s	10 s
Sobressinal (M_p)	0.2946 %	1 %
MESR	0.0103	0.0023
MRSR	0.0091	-
Variabilidade	0.1441	-

Fonte: AUTOR.

APÊNDICE B – Simulador da Planta Piloto de Neutralização de pH

Por um problema de praticidade é impossível desenvolver os Testes nº 1, nº 2 e nº 3 na planta real. O fato de trabalhar com algoritmos de aprendizado automático cria a necessidade de se ter uma grande quantidade de dados. Normalmente neste tipo de aplicações, os conjuntos têm mais de 1000 exemplos. O custo de tempo e dinheiro de repetir um experimento como apresentado na Figura B.6 é extremamente alto. Portanto, estas etapas são desenvolvidas em um ambiente simulado. O ambiente simulado da planta de pH é baseado no simulador obtido no trabalho de (GARCIA MARQUES, 2015). Os bons resultados desta dissertação e aproveitando o fato que o simulador permite alterar a resposta da planta colocando ruído facilmente foram as principais justificativas para seu uso.

O Teste nº 1 da Planta Piloto de Neutralização de pH é composto por cinco experimentos de identificação para cada uma das malhas. Basicamente, o procedimento foi excitar os modelos obtidos pela identificação da planta simulada em (GARCIA MARQUES, 2015) livre de ruído, com distintos sinais e alterar sua saída, adicionando distintos tipos de ruído. Desta forma, vão ser identificados modelos com uma boa variedade de dinâmicas. A Tabela B. 1 descreve alguma de suas características, destacando-se que o SNR foi calculado usando a variância.

Tabela B. 1- Características dos experimentos do Teste nº 1 na Planta de Neutralização de pH.

Experimento No.	Sinal de Excitação		SNR (dB)		Figura
	Nível	pH	Nível	pH	
1	Degrau	Degrau	∞	∞	Figura B.1
2	GBN	GBN	∞	∞	Figura B.2
3	Degrau	Degrau	2.05	-2.8	Figura B.3
4	GBN	GBN	16.84	19.59	Figura B.4
5	GBN	GBN	8.95	9.73	Figura B.5

Fonte: AUTOR.

Figura B.1- Experimentos no.1 de identificação da planta de Neutralização de pH.

Fonte: AUTOR.

Figura B.2- Experimentos no.2 de identificação da planta de Neutralização de pH.

Fonte: AUTOR.

Figura B.3-Experimentos no.3 de identificação da planta de Neutralização de pH.

Fonte: AUTOR.

Figura B.4- Experimentos no.4 de identificação da planta de Neutralização de pH.

Fonte: AUTOR.

Figura B.5- Experimentos no. 5 de identificação da planta de Neutralização de pH.

Fonte: AUTOR.

Para fazer o Teste n° 3 da planta de pH foram usados os mesmos modelos do teste anterior. Neste caso, do mesmo modo que no Apêndice A, se gerou uma malha de controle com o MPC com os valores de sintonia do Capítulo 4 e o processo é o modelo do simulador livre de ruído (também chamado modelo ideal). A Figura B.6 apresenta a resposta do simulador quando o MPC usa o modelo ideal.

Resposta malha de pH controlada pelo MPC com modelos Originais

(b)

Figura B.6- Resposta das malhas de controle da planta de Neutralização de pH usando o MPC com modelos ideais. (a) malha de nível e (b) malha de pH.

Fonte: AUTOR.

Tabela B.2- KPI das malhas de controle da planta de Neutralização de pH usando o MPC com modelos originais.

Malha	Harris	r_{wi}	IAU	MESR	MRSR	VAR	e_{ss}	$t_r(s)$	$t_s(s)$	$M_p(\%)$
Nível	3.32	20729	1047994	0.00042	$1.14e^{-5}$	0.0321	0	113	137	12.89
pH	3.64	9555	112304	0.00014	$7.91e^{-6}$	0.0242	0	147	197	0.14

Fonte: AUTOR.

Para normalizar os valores, emprega-se a equação (B.1). Neste caso, os valores correspondentes para cada um dos KPIs utilizados tanto na planta real como no benchmark são apresentados na Tabela B.3.

$$\bar{x} = \frac{x - x_{ref}}{x_{basis}} \quad (B.1)$$

Tabela B.3- Valores de referência das distintas malhas de controle para normalizar os KPIs.

KPI	Malha de nível				Malha de pH			
	Planta		pH benchmark		Planta		pH benchmark	
	x_{ref}	x_{basis}	x_{ref}	x_{basis}	x_{ref}	x_{basis}	x_{ref}	x_{basis}
$t_s(s)$	487	60	229	60	307	60	197	60
$M_p(\%)$	48	48	4.68	4.68	47	47	0,14	1
MESR	0.00565	0.00565	0.00107	0.00107	0.003725	0.003725	$6.56e^{-4}$	$6.56e^{-4}$
ess	0	0,1	0	0.1	0	0.01	0	0.01
r_{wi}	1488	1488	3774	3774	163.45	163.45	795	795

Fonte: AUTOR.

APÊNDICE C – Distribuição dos KPIs nas classes de treinamento

Tabela C. 1- Estatísticas dos KPI normalizados das classes catalogadas.

	Classes	Qtd	Ts		MESR		MRSR		Ess		rwi		Mp	
			μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
Nível	A	419	-0.0471	0.0765	0.0056	0.004	-0.0106	0.0063	0.0001	0.0008	0.0229	0.0052	-0.1103	0.0457
	B	138	0.6385	0.5645	0.083	0.0825	-0.0325	0.0532	0	0	0.1584	0.1566	-0.3696	0.4382
	C	228	2.6798	0.411	0.128	0.1684	0.0951	0.1274	0.0118	0.1787	0.0763	0.1525	-0.06	0.3047
	D	316	4.5488	1.141	0.1733	0.0736	0.2616	0.121	0.0006	0.0062	-0.0162	0.1466	0.5409	0.5544
	E	626	6.9067	0.5729	0.2768	0.0638	0.5604	0.1215	0	0.0008	-0.1711	0.0772	1.5517	0.6531
	F	317	9.2038	0.9076	0.4133	0.0986	0.8284	0.124	0.0002	0.0025	-0.2478	0.0836	2.5696	0.9162
	G	213	13.2643	2.3116	0.7144	0.6207	1.2259	0.2696	0.0523	0.7507	-0.2823	0.2287	3.9194	2.1431
	Z	687	12.7682	28.3927	204.128	369.40	2.9881	8.63	156.19	285.60	5.35e ⁺²²	3.02e ⁺²³	879.4	22468
pH	A	554	0.0578	0.2986	0.03510	0.04490	0.0042	0.0134	0.0783	0.240	-0.0006	0.039	0.3302	0.5573
	B	391	0.9225	0.7458	0.1409	0.0431	0.0117	0.0217	0.0173	0.1318	0.0178	0.0673	1.343	0.619
	C	478	3.1152	0.7201	0.1814	0.0946	0.0361	0.0621	0.0861	0.3143	-0.0048	0.1283	1.4195	0.9262
	D	445	4.1884	1.0398	0.2273	0.1958	0.1211	0.179	0.0058	0.0741	0.0153	0.1786	1.9059	1.2901
	E	254	5.7381	1.7837	0.4557	0.3271	0.2366	0.2144	0.0008	0.0107	-0.059	0.2113	1.5436	1.7318
	F	199	8.7796	1.3027	0.6982	0.2947	0.3408	0.1971	0.001	0.0099	-0.2824	0.1535	0.2377	1.201
	G	319	10.7105	2.314	0.837	0.2686	0.4654	0.2973	0.0017	0.0146	-0.2871	0.1694	0.7326	2.5839
	Z	290	17.07	18.23	83.67	298.75	5.031	41.72	515.1	1872.32	7.14e ⁺¹⁶	2.9e ⁺¹⁷	31.71	373.99
Clarke	A	863	0.4355	0.2376	0.1993	0.0362	0.0254	0.0455	0.0014	0.0377	0.2892	0.0913	0.4000	0.4203
	B	508	0.7172	0.5191	0.2571	0.0514	0.0379	0.0562	0.0073	0.1167	0.3861	0.1416	0.6418	0.4878
	C	880	1.5577	0.3734	0.4014	0.0834	0.036	0.0858	4.5e-05	0.0001	0.7219	0.283	0.2339	0.5307
	D	746	2.0135	0.5819	0.4893	0.2000	0.0696	0.1476	0.0059	0.1095	0.9255	0.556	0.722	1.5291
	E	497	6.1208	3.3077	1.3733	1.122	0.2552	0.3667	0.004	0.0285	5.2701	5.5013	2.9339	4.6668
	F	481	35.813	12.720	9.4561	3.2975	0.0597	0.9059	0.9978	3.8016	116.53	68.132	-0.241	0.6821
	G	487	74.910	15.069	108.553	275.209	77.658	664.93	1352.57	4477.22	1100.82	1294.2	1.1946	28.272
	Z	647	95.029	0.8171	1.98e ⁺²⁵	3.42e ⁺²⁶	1.60e ⁺²⁵	2.28e ⁺²⁶	6.52e ⁺²⁶	1.17e ⁺²⁸	1.37e ⁺¹³	3.47e ⁺¹⁴	-0.2186	1.1683

APÊNDICE D – Histogramas entre as entradas e o ID_{Norm}

Quadro D.1 Histogramas com a relação entre os resultados dos testes de hipóteses dos modelos e a qualidade do sistema de controle com MPC na planta de Clarke.

Quadro D.2 Histogramas com a relação entre os resultados dos MAMEs dos modelos e a qualidade do sistema de controle com MPC na planta de Clarke.

Quadro D.3 Histogramas com a relação entre os resultados das métricas de Dvurecenska (a), DTW' (b) e DTW (c) dos modelos e a qualidade do sistema de controle com MPC na planta de Clarke.

Fonte: AUTOR.

Quadro D.4 Histogramas com a relação entre os resultados dos critérios de seleção dos modelos e a qualidade do sistema de controle com MPC na planta de Clarke.

Fonte: AUTOR.

Quadro D.5 Histogramas com a relação entre os resultados dos testes de hipóteses dos modelos e a qualidade do sistema de controle com MPC na malha de nível.

Fonte: AUTOR.

Quadro D.6 Histogramas com a relação entre os resultados dos MAMEs dos modelos e a qualidade do sistema de controle com MPC na malha de nível.

Fonte: AUTOR.

Quadro D.9 Histogramas com a relação entre os resultados dos testes de hipóteses dos modelos e a qualidade do sistema de controle com MPC na malha de pH.

Quadro D.10 Histogramas com a relação entre os resultados dos MAMEs dos modelos e a qualidade do sistema de controle com MPC na malha de pH.

Fonte: AUTOR.

Quadro D.11 Histogramas com a relação entre os resultados das métricas de Dvurecenska (a), DTW' (b) e DTW (c) dos modelos e a qualidade do sistema de controle com MPC na malha de pH.

Fonte: AUTOR.

Quadro D.12 Histogramas com a relação entre os resultados dos critérios de seleção dos modelos e a qualidade do sistema de controle com MPC na malha de pH.

Fonte: AUTOR.

APÊNDICE E – Métodos Qualitativos de Validação

Os métodos qualitativos, também conhecidos como análise visual, são usados para comparar valores reais e valores simulados. Estes métodos são baseados em técnicas heurísticas usando o senso comum para analisar gráficos e chegar a uma conclusão sobre a validade do modelo. Os resultados obtidos são subjetivos e difíceis de quantificar. Métodos qualitativos são úteis em alguns cenários, especialmente quando se trata de identificar possíveis causas de erros de modelo. Eles também são robustos e fáceis de usar, razão pela qual eles são amplamente usados. As principais desvantagens destas técnicas são que não é possível quantificar o resultado das análises e a resposta muda, dependendo da pessoa, o que dificulta a segunda etapa da validação (KAT; ELS, 2012). Neste trabalho, os diferentes tipos de gráficos usados para realizar a análise visual foram o *boxplot* e o histograma.

O gráfico do tipo *boxplot* é visto na Figura E.1, onde cada caixa representa um conjunto de dados, a linha central vermelha representa a mediana; o fundo e a tampa da caixa simbolizam o quartil¹⁴ inferior (25%) e o quartil superior (75%), e os símbolos (+) indicam os *outliers*¹⁵ (MATHWORKS, 2006c). Os fios de bigode que se estendem verticalmente a partir da caixa indicam a variabilidade fora do quartil superior (adjacente superior) e do quartil inferior (adjacente inferior). A vantagem de mostrar os dados deste modo é que esta representação permite diferenciar facilmente os *outliers* e a distribuição dos dados ao se usar um grande número de dados.

Figura E. 1- Exemplo do gráfico Boxplot.

Fonte: (MATHWORKS, 2019c).

¹⁴ O quartil de um conjunto de amostras é um qualquer dos três valores que divide o conjunto ordenado de dados em quartos iguais e assim cada parte representa $\frac{1}{4}$ da amostra ou população.

¹⁵ Um *outlier* é considerado se a relação entre cada resíduo e variância dos resíduos seja maior do que 2 ou menor do que -2 (ROUSSEEUW; LEROY, 1987).

O histograma é essencialmente uma distribuição de frequências, em que os valores dos dados de origem agrupam-se em bandejas ou intervalos de classe; permitindo mostrar graficamente a distribuição dos diferentes elementos que compõem a série de dados observados. Nestes casos pode-se tentar aproximá-las mediante um modelo teórico de distribuição normal, logarítmico, exponencial, gaussiano, entre outros. Uma grande vantagem de usar os histogramas é que eles são úteis para grandes quantidades de dados observados (MORALES ALVARADO, 2017).

APÊNDICE F – Script de Matlab® para treinamento das Árvores de Decisão

Planta de Clarke

```
function [trainedClassifier, validationAccuracy] =
tree841Clarke(trainingData)
% [trainedClassifier, validationAccuracy] = tree841Clarke (trainingData)
% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%   trainingData: a table containing the same predictor and response
%   columns as imported into the app.
%
% Output:
%   trainedClassifier: a struct containing the trained classifier. The
%   struct contains various fields with information about the trained
%   classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions on new
%   data.
%
%   validationAccuracy: a double containing the accuracy in percent. In
%   the app, the History list displays this overall accuracy score for
%   each model.

% Extract predictors and response
% This code processes the data into the right shape for training the model.
inputTable = trainingData;
predictorNames = {'KS', 'AD', 'SW', 'WX', 'Fit', 'TIC', 'Willmott',
'Russell_Pr', 'Russell_Mr', 'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D',
'CodExp', 'N_nAIC', 'N_BIC', 'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts',
'V_Mp', 'V_MESR', 'V_MRSR', 'V_Var', 'V_Difs'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false];

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = predictors.Properties.VariableNames([false false
false false false false true true false false true true false false false
true false false false true true true false false false]);
predictors = predictors(:, includedPredictorNames);
isCategoricalPredictor = isCategoricalPredictor([false false false false
false false true true false false true true false false false true false
false false false true true true false false false false]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    predictors, ...
    response, ...
```

```

    'SplitCriterion', 'gdi', ...
    'MaxNumSplits', 370, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
featureSelectionFcn = @(x) x(:, includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
trainedClassifier.predictFcn = @(x)
treePredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'AD', 'Anova', 'CodExp', 'DTW',
'DTW_D', 'Dvure', 'Fit', 'ITAE', 'KS', 'N_BIC', 'N_FPE', 'N_nAIC',
'Russell_Mr', 'Russell_Pr', 'SG_Mr', 'SW', 'TIC', 'V_Difs', 'V_MESR',
'V_MRSR', 'V_Mp', 'V_Tr', 'V_Ts', 'V_Var', 'V_ess', 'WX', 'Willmott'};
trainedClassifier.ClassificationTree = classificationTree;
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'KS', 'AD', 'SW', 'WX', 'Fit', 'TIC', 'Willmott',
'Russell_Pr', 'Russell_Mr', 'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D',
'CodExp', 'N_nAIC', 'N_BIC', 'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts',
'V_Mp', 'V_MESR', 'V_MRSR', 'V_Var', 'V_Difs'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false];

% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.3);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = trainingPredictors.Properties.VariableNames([false
false false false false false true true false false true true false false
false true false false false false true true true false false false
false]);
trainingPredictors = trainingPredictors(:, includedPredictorNames);
trainingIsCategoricalPredictor = trainingIsCategoricalPredictor([false
false false false false false true true false false true true false false
false true false false false false true true true false false false
false]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    trainingPredictors, ...
    trainingResponse, ...
    'SplitCriterion', 'gdi', ...
    'MaxNumSplits', 370, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

```

```

% Create the result struct with predict function
featureSelectionFcn = @(x) x(:,includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
validationPredictFcn = @(x) treePredictFcn(featureSelectionFcn(x));

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

Planta de Neutralização de pH

Malha de Nível

```

function [trainedClassifier, validationAccuracy] =
treeacc879H(trainingData)
% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData)
% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%   trainingData: a table containing the same predictor and response
%   columns as imported into the app.
%
% Output:
%   trainedClassifier: a struct containing the trained classifier. The
%   struct contains various fields with information about the trained
%   classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions on new
%   data.
%
%   validationAccuracy: a double containing the accuracy in percent. In
%   the app, the History list displays this overall accuracy score for
%   each model.
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'AD', 'SW', 'WX', 'Fit', 'TIC', 'Willmott', 'Russell_Pr',
'Russell_Mr', 'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D', 'CodExp',
'N_nAIC', 'N_BIC', 'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts', 'V_Mp',
'V_MESR', 'V_MRSR', 'V_Var'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;

```

```
isCategoricalPredictor = [false, false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false, false, false];
```

```
% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = predictors.Properties.VariableNames([false false
false false true true true false false false true false false false false
false false false false true true true false false false]);
predictors = predictors(:,includedPredictorNames);
isCategoricalPredictor = isCategoricalPredictor([false false false false
true true true false false false true false false false false false
false false true true true false false false]);
```

```
% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    predictors, ...
    response, ...
    'SplitCriterion', 'twoing', ...
    'MaxNumSplits', 348, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);
```

```
% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
featureSelectionFcn = @(x) x(:,includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
trainedClassifier.predictFcn = @(x)
treePredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));
```

```
% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'AD', 'Anova', 'CodExp', 'DTW',
'DTW_D', 'Dvure', 'Fit', 'ITAE', 'N_BIC', 'N_FPE', 'N_nAIC', 'Russell_Mr',
'Russell_Pr', 'SG_Mr', 'SW', 'TIC', 'V_MESR', 'V_MRSR', 'V_Mp', 'V_Tr',
'V_Ts', 'V_Var', 'V_ess', 'WX', 'Willmott'};
trainedClassifier.ClassificationTree = classificationTree;
```

```
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'AD', 'SW', 'WX', 'Fit', 'TIC', 'Willmott', 'Russell_Pr',
'Russell_Mr', 'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D', 'CodExp',
'N_nAIC', 'N_BIC', 'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts', 'V_Mp',
'V_MESR', 'V_MRSR', 'V_Var'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false, false, false];
```

```
% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.3);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;
```

```
% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
```

```

includedPredictorNames = trainingPredictors.Properties.VariableNames([false
false false false true true true false false false true false false false
false false false false false true true true false false false]);
trainingPredictors = trainingPredictors(:,includedPredictorNames);
trainingIsCategoricalPredictor = trainingIsCategoricalPredictor([false
false false false true true true false false false true false false false
false false false false false true true true false false false]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    trainingPredictors, ...
    trainingResponse, ...
    'SplitCriterion', 'twoing', ...
    'MaxNumSplits', 348, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
featureSelectionFcn = @(x) x(:,includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
validationPredictFcn = @(x) treePredictFcn(featureSelectionFcn(x));

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

Malha de pH

```

function [trainedClassifier, validationAccuracy] =
treeacc899pHClassifier(trainingData)
% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData)
% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%     trainingData: a table containing the same predictor and response
%                   columns as imported into the app.
%
% Output:
%     trainedClassifier: a struct containing the trained classifier. The
%                       struct contains various fields with information about the trained
%                       classifier.
%
%     trainedClassifier.predictFcn: a function to make predictions on new
%                                   data.

```



```

%
% validationAccuracy: a double containing the accuracy in percent. In
% the app, the History list displays this overall accuracy score for
% each model.

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Fit', 'TIC', 'Willmott', 'Russell_Pr', 'Russell_Mr',
'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D', 'CodExp', 'N_nAIC', 'N_BIC',
'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts', 'V_Mp', 'V_MESR', 'V_MRSR',
'V_Var', 'V_Difs'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false,
false, false, false, false, false, false];

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = predictors.Properties.VariableNames([false true
true true false false false true false false false false false false
false true true true false false false true]);
predictors = predictors(:, includedPredictorNames);
isCategoricalPredictor = isCategoricalPredictor([false true true true false
false false true false false false false false false false true true
true false false false true]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    predictors, ...
    response, ...
    'SplitCriterion', 'deviance', ...
    'MaxNumSplits', 237, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
featureSelectionFcn = @(x) x(:, includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
trainedClassifier.predictFcn = @(x)
treePredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Anova', 'CodExp', 'DTW', 'DTW_D',
'Dvure', 'Fit', 'ITAE', 'N_BIC', 'N_FPE', 'N_nAIC', 'Russell_Mr',
'Russell_Pr', 'SG_Mr', 'TIC', 'V_Difs', 'V_MESR', 'V_MRSR', 'V_Mp', 'V_Tr',
'V_Ts', 'V_Var', 'V_ess', 'Willmott'};
trainedClassifier.ClassificationTree = classificationTree;

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'Fit', 'TIC', 'Willmott', 'Russell_Pr', 'Russell_Mr',
'SG_Mr', 'Anova', 'Dvure', 'DTW', 'DTW_D', 'CodExp', 'N_nAIC', 'N_BIC',

```

```

'N_FPE', 'ITAE', 'V_ess', 'V_Tr', 'V_Ts', 'V_Mp', 'V_MESR', 'V_MRSR',
'V_Var', 'V_Difs'});
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false,
false, false, false, false, false, false, false, false, false,
false, false, false, false, false, false];

% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.3);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = trainingPredictors.Properties.VariableNames([false
true true true false false false true false false false false false
false false true true true false false true]);
trainingPredictors = trainingPredictors(:, includedPredictorNames);
trainingIsCategoricalPredictor = trainingIsCategoricalPredictor([false true
true true false false false true false false false false false false
false true true true false false false true]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
classificationTree = fitctree(...
    trainingPredictors, ...
    trainingResponse, ...
    'SplitCriterion', 'deviance', ...
    'MaxNumSplits', 237, ...
    'Surrogate', 'off', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
featureSelectionFcn = @(x) x(:, includedPredictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
validationPredictFcn = @(x) treePredictFcn(featureSelectionFcn(x));

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

APÊNDICE G – Script de Matlab® para treinamento das Redes Neurais Artificiais

Planta de Clarke

```
function [net_Clarke,AUC_Clarke]=ANN_Clarke(New_TR,Target)
% Solve a Pattern Recognition Problem with a Neural Network
%   New_TR - input data.
%   Target - target data.

x = New_TR';
t = Target';

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainscg'; % Scaled conjugate gradient backpropagation.

% Create a Pattern Recognition Network
hiddenLayerSize = 45;
net_Clarke = patternnet(hiddenLayerSize, trainFcn);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net_Clarke.input.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivision
net_Clarke.divideFcn = 'dividerand'; % Divide data randomly
net_Clarke.divideMode = 'sample'; % Divide up every sample
net_Clarke.divideParam.trainRatio = 70/100;
net_Clarke.divideParam.valRatio = 5/100;
net_Clarke.divideParam.testRatio = 25/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net_Clarke.performFcn = 'crossentropy'; % Cross-Entropy

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net_Clarke.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
    'plotconfusion','plotroc'};

% Train the Network
[net_Clarke,tr] = train(net_Clarke,x,t);

% Test the Network
y = net_Clarke(x);
e = gsubtract(t,y);
performance = perform(net_Clarke,t,y)
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);
```

```

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(net_Clarke,trainTargets,y)
valPerformance = perform(net_Clarke,valTargets,y)
testPerformance = perform(net_Clarke,testTargets,y)

% View the Network
view(net_Clarke)

% Plots

yvalTargets = net_Clarke(x(:,tr.testInd));
figure, plotconfusion(t(:,tr.testInd),yvalTargets)
[tp, fpr, ~]=roc(t(:,tr.testInd),yvalTargets);
AUC_Clarke=aucVD(fpr,tp);

```

Malha de Nível

```

function [net_H,AUC_H]=ANN_H(New_TR_H,Target_H)
% Solve a Pattern Recognition Problem with a Neural Network
%
%   New_TR_H - input data.
%   Target_H - target data.

x = New_TR_H';
t = Target_H';

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainscg'; % Scaled conjugate gradient backpropagation.

% Create a Pattern Recognition Network
hiddenLayerSize = 97;
net_H = patternnet(hiddenLayerSize, trainFcn);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net_H.input.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivision
net_H.divideFcn = 'dividerand'; % Divide data randomly
net_H.divideMode = 'sample'; % Divide up every sample
net_H.divideParam.trainRatio = 70/100;
net_H.divideParam.valRatio = 5/100;
net_H.divideParam.testRatio = 25/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net_H.performFcn = 'crossentropy'; % Cross-Entropy

% Choose Plot Functions

```

```

% For a list of all plot functions type: help nnplot
net_H.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
    'plotconfusion','plotroc'};

% Train the Network
[net_H,tr] = train(net_H,x,t);

% Test the Network
y = net_H(x);
e = gsubtract(t,y);
performance = perform(net_H,t,y)
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(net_H,trainTargets,y)
valPerformance = perform(net_H,valTargets,y)
testPerformance = perform(net_H,testTargets,y)

% View the Network
view(net_H)

% Plots
yvalTargets = net_H(x(:,tr.testInd));
figure, plotconfusion(t(:,tr.testInd),yvalTargets)
    [tpr,fpr,~]=roc(t(:,tr.testInd),yvalTargets);
AUC_H=aucVD(fpr,tpr);

end

```

Malha de pH

```

function [net_pH,AUC_pH]=ANN_pH(New_TR_PH,Target_PH)
% Solve a Pattern Recognition Problem with a Neural Network
% Script generated by Neural Pattern Recognition app
% Created 18-May-2020 11:47:26
%
% This script assumes these variables are defined:
%
%   New_TR_PH - input data.
%   Target_PH - target data.

x = New_TR_PH';
t = Target_PH';

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainscg'; % Scaled conjugate gradient backpropagation.

% Create a Pattern Recognition Network
hiddenLayerSize = 98;

```

```

net_pH = patternnet(hiddenLayerSize, trainFcn);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net_pH.input.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivision
net_pH.divideFcn = 'dividerand'; % Divide data randomly
net_pH.divideMode = 'sample'; % Divide up every sample
net_pH.divideParam.trainRatio = 70/100;
net_pH.divideParam.valRatio = 5/100;
net_pH.divideParam.testRatio = 25/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net_pH.performFcn = 'crossentropy'; % Cross-Entropy

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net_pH.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
    'plotconfusion','plotroc'};

% Train the Network
[net_pH,tr] = train(net_pH,x,t);

% Test the Network
y = net_pH(x);
e = gsubtract(t,y);
performance = perform(net_pH,t,y)
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(net_pH,trainTargets,y)
valPerformance = perform(net_pH,valTargets,y)
testPerformance = perform(net_pH,testTargets,y)

% View the Network
view(net_pH)

% Plots
yvalTargets = net_pH(x(:,tr.testInd));
figure, plotconfusion(t(:,tr.testInd),yvalTargets)
[tpr,fpr,~]=roc(t(:,tr.testInd),yvalTargets);
AUC_pH=aucVD(fpr,tpr);

end

```

APÊNDICE H – Script de Matlab® para treinamento das Máquina de Vetores de Suporte

Planta de Clarke

```
function [trainedClassifier, validationAccuracy] =
svmacc819Clarke(trainingData)
% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData)
% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%   trainingData: a table containing the same predictor and response
%   columns as imported into the app.
%
% Output:
%   trainedClassifier: a struct containing the trained classifier. The
%   struct contains various fields with information about the trained
%   classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions on new
%   data.
%
%   validationAccuracy: a double containing the accuracy in percent. In
%   the app, the History list displays this overall accuracy score for
%   each model.
%
inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 50.5582807023888, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    predictors, ...
    response, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x)
svmPredictFcn(predictorExtractionFcn(x));
```

```

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Dvure', 'Russell_Pr', 'TIC',
'V_Mp', 'V_Tr', 'V_Ts', 'Willmott'};
trainedClassifier.ClassificationSVM = classificationSVM;

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.32);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 50.5582807023888, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    trainingPredictors, ...
    trainingResponse, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
svmPredictFcn = @(x) predict(classificationSVM, x);
validationPredictFcn = @(x) svmPredictFcn(x);

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

Malha de Nível

```

function [trainedClassifier, validationAccuracy] = svmacc915H(trainingData)
% [trainedClassifier, validationAccuracy] = svmacc915H(trainingData)

```



```

% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%     trainingData: a table containing the same predictor and response
%     columns as imported into the app.
%
% Output:
%     trainedClassifier: a struct containing the trained classifier. The
%     struct contains various fields with information about the trained
%     classifier.
%
%     trainedClassifier.predictFcn: a function to make predictions on new
%     data.
%
%     validationAccuracy: a double containing the accuracy in percent. In
%     the app, the History list displays this overall accuracy score for
%     each model.
%
% Use the code to train the model with new data. To retrain your
% classifier, call the function from the command line with your original
% data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
%     [trainedClassifier, validationAccuracy] = trainClassifier(T)

inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = predictors.Properties.VariableNames([true false
true false true false true]);
predictors = predictors(:, includedPredictorNames);
isCategoricalPredictor = isCategoricalPredictor([true false true false true
false true]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 999.296388229406, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    predictors, ...
    response, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);

```

```

featureSelectionFcn = @(x) x(:,includedPredictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x)
svmPredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Dvure', 'Russell_Pr', 'TIC',
'V_Mp', 'V_Tr', 'V_Ts', 'Willmott'};
trainedClassifier.ClassificationSVM = classificationSVM;

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.32);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = trainingPredictors.Properties.VariableNames([true
false true false true false true]);
trainingPredictors = trainingPredictors(:,includedPredictorNames);
trainingIsCategoricalPredictor = trainingIsCategoricalPredictor([true false
true false true false true]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 999.296388229406, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    trainingPredictors, ...
    trainingResponse, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
featureSelectionFcn = @(x) x(:,includedPredictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
validationPredictFcn = @(x) svmPredictFcn(featureSelectionFcn(x));

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);

```

```
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);
```

Malha de pH

```
function [trainedClassifier, validationAccuracy] =
svmmacc910pH(trainingData)
% [trainedClassifier, validationAccuracy] = svmmacc910pH(trainingData)
% returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
% Input:
%   trainingData: a table containing the same predictor and response
%   columns as imported into the app.
%
% Output:
%   trainedClassifier: a struct containing the trained classifier. The
%   struct contains various fields with information about the trained
%   classifier.
%
%   trainedClassifier.predictFcn: a function to make predictions on new
%   data.
%
%   validationAccuracy: a double containing the accuracy in percent. In
%   the app, the History list displays this overall accuracy score for
%   each model.
%
% Use the code to train the model with new data. To retrain your
% classifier, call the function from the command line with your original
% data or new data as the input argument trainingData.
%
inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = predictors.Properties.VariableNames([false true
true false true false false]);
predictors = predictors(:, includedPredictorNames);
isCategoricalPredictor = isCategoricalPredictor([false true true false true
false false]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
'KernelFunction', 'gaussian', ...
'PolynomialOrder', [], ...
```

```

    'KernelScale', 'auto', ...
    'BoxConstraint', 693.1246241701939, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    predictors, ...
    response, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
featureSelectionFcn = @(x) x(:,includedPredictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
trainedClassifier.predictFcn = @(x)
svmPredictFcn(featureSelectionFcn(predictorExtractionFcn(x)));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Dvure', 'Russell_Pr', 'TIC',
'V_Mp', 'V_Tr', 'V_Ts', 'Willmott'};
trainedClassifier.ClassificationSVM = classificationSVM;

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = trainingData;
predictorNames = {'TIC', 'Willmott', 'Russell_Pr', 'Dvure', 'V_Tr', 'V_Ts',
'V_Mp'};
predictors = inputTable(:, predictorNames);
response = inputTable.ID_norm;
isCategoricalPredictor = [false, false, false, false, false, false, false];

% Set up holdout validation
cvp = cvpartition(response, 'Holdout', 0.32);
trainingPredictors = predictors(cvp.training, :);
trainingResponse = response(cvp.training, :);
trainingIsCategoricalPredictor = isCategoricalPredictor;

% Data transformation: Select subset of the features
% This code selects the same subset of features as were used in the app.
includedPredictorNames = trainingPredictors.Properties.VariableNames([false
true true false true false false]);
trainingPredictors = trainingPredictors(:,includedPredictorNames);
trainingIsCategoricalPredictor = trainingIsCategoricalPredictor([false true
true false true false false]);

% Train a classifier
% This code specifies all the classifier options and trains the classifier.
template = templateSVM(...
    'KernelFunction', 'gaussian', ...
    'PolynomialOrder', [], ...
    'KernelScale', 'auto', ...
    'BoxConstraint', 693.1246241701939, ...
    'Standardize', false);
classificationSVM = fitcecoc(...
    trainingPredictors, ...
    trainingResponse, ...
    'Learners', template, ...
    'Coding', 'onevsone', ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8]);

```

```

% Create the result struct with predict function
featureSelectionFcn = @(x) x(:,includedPredictorNames);
svmPredictFcn = @(x) predict(classificationSVM, x);
validationPredictFcn = @(x) svmPredictFcn(featureSelectionFcn(x));

% Add additional fields to the result struct

% Compute validation predictions
validationPredictors = predictors(cvp.test, :);
validationResponse = response(cvp.test, :);
[validationPredictions, validationScores] =
validationPredictFcn(validationPredictors);

% Compute validation accuracy
correctPredictions = (validationPredictions == validationResponse);
isMissing = isnan(validationResponse);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions);

```

APÊNDICE I – Representação gráfica das Árvores de Decisão treinadas

Figura I.1 Representação gráfica da DT da planta de Clarke.

Fonte: AUTOR.

Figura I.2- Representação gráfica da DT da malha de nível da Planta Piloto de Neutralização de pH.

Fonte: AUTOR.

APÊNDICE J – Tabela com os modelos testados na Planta Piloto de Neutralização de pH

Tabela J.1- Funções de transferência dos modelos testados na planta de pH.

Nível		pH	
Modelo No.	Modelo polinomial não paramétrico	Modelo No.	Modelo polinomial não paramétrico
19	$A(z) = 1 - 1.98 z^{-1} + 0.9805 z^{-2} - 3.653e-12 z^{-3}$ $B(z) = 6.507e-15 z^{-1} + 0.004577 z^{-2} - 0.004258 z^{-3}$	2350	$A(z) = 1 - 1.809 z^{-1} + 0.6698 z^{-2} + 0.1395 z^{-3}$ $B(z) = 0.003844 z^{-1} + 0.03359 z^{-2} - 0.03732 z^{-3}$ $C(z) = 1 - 1.506 z^{-1} + 0.4202 z^{-2} + 0.09706 z^{-3} + 0.01005 z^{-4} - 0.00737 z^{-5}$
1930	$A(z) = 1 - 2.873 z^{-1} + 2.749 z^{-2} - 0.8756 z^{-3}$ $B(z) = -0.02641 z^{-1} + 0.1023 z^{-2} - 0.143 z^{-3} + 0.08542 z^{-4} - 0.01832 z^{-5}$ $C(z) = 1 - 2.795 z^{-1} + 2.59 z^{-2} - 0.7735 z^{-3} - 0.04159 z^{-4} + 0.01963 z^{-5}$	2347	$A(z) = 1 - 1.956 z^{-1} + 0.9548 z^{-2} + 0.0007866 z^{-3}$ $B(z) = 0.007329 z^{-1} + 0.02104 z^{-2} - 0.02829 z^{-3}$ $C(z) = 1 - 1.652 z^{-1} + 0.6637 z^{-2}$
3117	$B(z) = 0.221 z^{-30} - 0.4465 z^{-31} + 0.05723 z^{-32} + 0.3423 z^{-33} - 0.174 z^{-34}$ $F(z) = 1 - 1.892 z^{-1} - 0.08176 z^{-2} + 1.693 z^{-3} - 0.5704 z^{-4} - 0.1486 z^{-5}$	2344	$A(z) = 1 - 1.753 z^{-1} + 0.5589 z^{-2} + 0.1947 z^{-3}$ $B(z) = 0.004125 z^{-1} + 0.03438 z^{-2} - 0.0384 z^{-3}$ $C(z) = 1 - 1.45 z^{-1} + 0.3282 z^{-2} + 0.1352 z^{-3}$
3148	$B(z) = -0.02121 z^{-1} - 0.03414 z^{-2} + 0.07368 z^{-3} + 0.1761 z^{-4} + 0.02611 z^{-5} - 0.2257 z^{-6} - 0.2637 z^{-7} - 0.001123 z^{-8} + 0.2501 z^{-9} + 0.1708 z^{-10} + 0.009805 z^{-11} - 0.1264 z^{-12} - 0.2166 z^{-13} + 0.06218 z^{-14} + 0.2723 z^{-15} - 0.00378 z^{-16} - 0.1776 z^{-17} - 0.03623 z^{-18} + 0.04729 z^{-19} + 0.01841 z^{-20}$ $F(z) = 1 + 2.022 z^{-1} - 0.9468 z^{-2} - 4.474 z^{-3} - 2.279 z^{-4} + 1.534 z^{-5} + 2.796 z^{-6} + 2.272 z^{-7} + 0.04115 z^{-8} - 1.355 z^{-9} - 0.6117 z^{-10}$	2395	$A(z) = 1 - 0.9571 z^{-1} - 0.9926 z^{-2} + 0.9499 z^{-3}$ $B(z) = 0.004849 z^{-1} + 0.03341 z^{-2} - 0.01206 z^{-3} - 0.01871 z^{-4} + 0.0009291 z^{-5} - 0.02553 z^{-6} + 0.0162 z^{-7} + 0.008411 z^{-8} - 0.007417 z^{-9} + 0.0001417 z^{-10}$ $C(z) = 1 - 0.6537 z^{-1} - 0.984 z^{-2} + 0.6624 z^{-3} + 0.0018 z^{-4} + 0.0001983 z^{-5}$
47	$A(z) = 1 - 2.073 z^{-1} + 1.076 z^{-2} - 0.0003971 z^{-3} + 0.075 z^{-4} - 0.07757 z^{-5}$ $B(z) = 0.0001679 z^{-15}$	160	$A(z) = 1 - 1.11 z^{-1} - 0.2721 z^{-2} - 0.009148 z^{-3} + 0.3917 z^{-4} + 5.393e-12 z^{-5} - 7.3e-12 z^{-6} + 3.569e-12 z^{-7} - 3.609e-12 z^{-8} + 3.549e-12 z^{-9} - 1.079e-12 z^{-10}$ $B(z) = 3.879e-14 z^{-1} + 0.03831 z^{-2} - 0.005978 z^{-3} - 0.01612 z^{-4} - 0.01573 z^{-5}$
158	$A(z) = 1 - 2.054 z^{-1} + 1.057 z^{-2} - 0.0002894 z^{-3} - 0.0002724 z^{-4} - 0.0002568 z^{-5} - 0.0002426 z^{-6} - 0.0002297 z^{-7} - 0.0002179 z^{-8} + 0.04414 z^{-9} - 0.04601 z^{-10}$ $B(z) = 0.0001559 z^{-15} + 8.961e-07 z^{-16}$	2428	$A(z) = 1 - 1.964 z^{-1} + 0.9637 z^{-2}$ $B(z) = 0.01226 z^{-1} + 0.008035 z^{-2} - 0.02029 z^{-3}$ $C(z) = 1 - 1.664 z^{-1} + 0.6695 z^{-2} + 0.0005434 z^{-3} + 0.00985 z^{-4} - 0.01218 z^{-5} + 0.005302 z^{-6} - 0.005758 z^{-7} + 0.009858 z^{-8} + 0.0101 z^{-9} - 0.01992 z^{-10}$

1932	$A(z) = 1 - 2.914 z^{-1} + 2.829 z^{-2} - 0.9153 z^{-3}$ $B(z) = 0.1056 z^{-30} - 0.3111 z^{-31} + 0.3274 z^{-32} - 0.1427 z^{-33} + 0.02082 z^{-34}$ $C(z) = 1 - 2.804 z^{-1} + 2.607 z^{-2} - 0.7774 z^{-3} - 0.04954 z^{-4} + 0.02348 z^{-5}$	2443	$A(z) = 1 - 1.965 z^{-1} + 0.9645 z^{-2}$ $B(z) = 0.03187 z^{-1} - 0.03186 z^{-2}$ $C(z) = 1 - 1.665 z^{-1} + 0.6701 z^{-2} + 0.0005358 z^{-3} + 0.009832 z^{-4} - 0.01204 z^{-5} + 0.005149 z^{-6} - 0.005547 z^{-7} + 0.009809 z^{-8} + 0.009506 z^{-9} - 0.01914 z^{-10}$
962	$A(z) = 1 - 0.9864 z^{-1}$ $B(z) = 0.00247 z^{-15} + 0.005572 z^{-16} + 0.0002619 z^{-17} + 0.0002584 z^{-18} + 0.0002555 z^{-19} + 0.0002519 z^{-20} + 0.0002489 z^{-21} + 0.0002453 z^{-22} + 0.002677 z^{-23} + 0.00258 z^{-24}$ $C(z) = 1 + 0.9734 z^{-1}$	11	$A(z) = 1 - 0.9899 z^{-1}$ $B(z) = 0.02458 z^{-15} - 0.0007336 z^{-16} - 0.0006989 z^{-17} - 0.0006657 z^{-18} - 0.01262 z^{-19}$
1131	$A(z) = 1 - 1.984 z^{-1} + 0.3322 z^{-2} + 0.9522 z^{-3} + 0.03508 z^{-4} - 0.3357 z^{-5}$ $B(z) = 3.656e-06 z^{-30}$ $C(z) = 1 + 0.06384 z^{-1} - 0.5884 z^{-2} - 0.3194 z^{-3}$	14	$A(z) = 1 - 0.994 z^{-1}$ $B(z) = 0.02306 z^{-15} - 0.0008189 z^{-16} - 0.0007803 z^{-17} - 0.0007435 z^{-18} - 0.0007084 z^{-19} - 0.000675 z^{-20} - 0.0006431 z^{-21} - 0.0006128 z^{-22} - 0.0005839 z^{-23} - 0.01163 z^{-24}$
1142	$A(z) = 1 - 2.985 z^{-1} + 2.97 z^{-2} - 0.9845 z^{-3} - 0.002349 z^{-4} + 0.001336 z^{-5}$ $B(z) = -4.05e-06 z^{-15} - 4.918e-06 z^{-16} + 9.277e-06 z^{-17}$ $C(z) = 1 - 0.9371 z^{-1}$	464	$A(z) = 1 - 0.9987 z^{-1}$ $B(z) = 0.001179 z^{-15}$ $C(z) = 1 + 1.026 z^{-1} + 1.015 z^{-2} + 0.9742 z^{-3} + 0.9063 z^{-4} + 0.8137 z^{-5} + 0.7003 z^{-6} + 0.5706 z^{-7} + 0.4297 z^{-8} + 0.2833 z^{-9} + 0.1374 z^{-10}$
1145	$A(z) = 1 - 1.854 z^{-1} - 0.09183 z^{-2} + 1.443 z^{-3} - 0.1954 z^{-4} - 0.3024 z^{-5}$ $B(z) = -1.127e-05 z^{-15} + 1.534e-06 z^{-16} + 1.147e-05 z^{-17}$ $C(z) = 1 + 0.1938 z^{-1} - 0.7464 z^{-2} - 0.2884 z^{-3}$	465	$A(z) = 1 - 1.001 z^{-1}$ $B(z) = -0.000873 z^{-30}$ $C(z) = 1 + 1.023 z^{-1} + 1.017 z^{-2} + 0.98 z^{-3} + 0.9147 z^{-4} + 0.8228 z^{-5} + 0.7093 z^{-6} + 0.5786 z^{-7} + 0.4362 z^{-8} + 0.2881 z^{-9} + 0.1405 z^{-10}$

Fonte: AUTOR.