



**Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica**

**GPU-Based Embedded Monitoring
for Fault Detection and Diagnosis
in Time-Varying Nonlinear
Systems - A Wind Turbine Case Study**

Marlon Marques Soudré

**TESE DE DOUTORADO
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS MECATRÔNICOS**

Brasília
2023

**Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica**

**Monitoramento Embarcado Baseado em GPU
para Detecção e Diagnóstico de Falhas em
Sistemas Não Lineares Variantes no Tempo -
Um Estudo de Caso de Turbina Eólica**

Marlon Marques Soudré

Tese de Doutorado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Doutor

Orientador: Prof. Dr. Carlos Humberto Llanos Quintero

Coorientador: Prof. Dr. Helon Vicente Hultmann Ayala

Brasília

2023

S769g Soudré, Marlon Marques .
GPU-Based Embedded Monitoring for Fault Detection and
Diagnosis in Time-Varying Nonlinear Systems - A Wind Turbine
Case Study / Marlon Marques Soudré; orientador Carlos Hum-
berto Llanos Quintero; coorientador Helon Vicente Hultmann
Ayala. -- Brasília, 2023.
204 p.

Tese de Doutorado (Programa de Pós-Graduação em Sistemas
Mecatrônicos) -- Universidade de Brasília, 2023.

1. Monitoramento de Sistemas Variantes no Tempo. 2. Projeto
de Sistemas Embarcados. 3. Detecção e Diagnóstico de Falhas.
4. Falhas em Turbinas Eólicas. I. Quintero, Carlos Humberto
Llanos, orient. II. Ayala, Helon Vicente Hultmann, coorient. III.
Título

**Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica**

**GPU-Based Embedded Monitoring
for Fault Detection and Diagnosis
in Time-Varying Nonlinear
Systems - A Wind Turbine Case Study**

Marlon Marques Soudré

Tese de Doutorado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Doutor

Trabalho aprovado. Brasília, 07 de Dezembro de 2023:

**Prof. Dr. Carlos Humberto Llanos
Quintero, UnB/FT/ENM**
Coorientador

**Prof. Dr. Helon Vicente Hultmann Ayala,
PuC-Rio/DEM**
Examinador externo

**Prof. Dr. ALVARO BERNAL NOREÑA,
Universidad del Valle/EIEE**
Examinador externo

**Prof. Dr. Edward David Moreno Ordonez,
IFS/DCOMP**
Examinador externo

**Prof. Dr. Leandro dos Santos Coelho
PUCPR/CCET/PPGEPS**
Examinador externo

Brasília
2023

Acknowledgements

To God, the author and preserver of life, who has always strengthened and sustained me in his infinite love. To you be all the honor and glory.

To my dear wife Marcella, for her patience and companionship on good and bad days throughout the years. We have closed another cycle together. May the next ones come!

To my parents, Baltazar and Ana Maria, in honor and gratitude, for their support and efforts so that I could get this far. This is just another fruit of what they have sown.

To my advisors, Prf. Carlos Llanos and Prf. Helon Ayala, for their teaching, guidance and encouragement throughout this work. You have my respect and admiration.

To the University of Brasilia for welcoming me and for the opportunities offered.

Finally, I would like to thank all those who, in some way, contributed to this achievement.

“Ó profundidade da riqueza, tanto da sabedoria como do conhecimento de Deus! Quão insondáveis são os seus juízos, e quão inescrutáveis, os seus caminhos! Quem, pois, conheceu a mente do Senhor? Ou quem foi o seu conselheiro? Ou quem primeiro deu a ele para que lhe venha a ser restituído? Porque dele, e por meio dele, e para ele são todas as coisas. A ele, pois, a glória eternamente. Amém!”
(Romanos 11:33-36)

Abstract

Monitoring non-linear and time-varying systems to detect and diagnose faults is not trivial, especially when applied to embedded systems and their constraints. Despite the challenges, the development and popularization of technologies such as the *Internet of Things* (IoT), machine learning tools, and the Edge Computing paradigm have increased interest in the subject. However, there are still few works in this scenario, and there is still a gap to be filled in terms of solutions that are viable to be embedded. With this in mind, this paper proposes contributions based on GPUs to help fill this gap in the literature. Firstly, a strategy is proposed for embedded systems based on GPUs to identify nonlinear and black-box systems. More precisely, a parallelization strategy was developed for the *Forward Regression Orthogonal Least Squares* (FROLS) algorithm to select parsimonious, linear, and non-linear autoregressive models. Next, solutions for fault detection and classification are presented, mapped on GPUs and based on the analysis of the parameters of the identified model, implemented with control chart tools and *Support Vector Machines* (SVM), respectively. Finally, the solutions above were combined, and a monitoring algorithm for fault detection and diagnosis (SWRLS_SVM_FROLS) was established using the recursive estimation of the model parameters within a *Slide Window Recursive Least Squares* (SWRLS). When validated with real data from a wind turbine blade subjected to temperature variation and failures due to ice accumulation and crack formation at different scales, the solutions proved viable, showing the correct detection and classification of faults, as well as computational gains and energy savings, which can reach 24.5x and 22x respectively in the case of system identification/re-identification. It is worth noting that this case study is topical and relevant, combining the characteristics of a time-varying nonlinear system due to faults and environmental factors and operating in the edge computing paradigm, essential especially for offshore installations. In this sense, the chosen case study explores the proposed solution, showing the viability of GPU-based embedded monitoring for detecting and diagnosing faults in nonlinear time-varying systems.

Keywords: Time-varying Systems Monitoring. Embedded System Design. Fault Detection and Diagnosis. Wind Turbine Faults.

Resumo

O monitoramento de sistemas não lineares e variantes no tempo visando detectar e diagnosticar falhas não é uma tarefa trivial, sobretudo quando aplicado a sistemas embarcados e suas restrições. Apesar dos desafios, o desenvolvimento e popularização de tecnologias como *Internet of Things* (IoT), ferramentas de aprendizagem de máquina e o paradigma de computação na borda (Edge Computing), tem aumentado o interesse no tema. No entanto, ainda são poucos os trabalhos inseridos nesse cenário, havendo um espaço a ser preenchido no que tange a soluções que sejam viáveis de serem embarcadas. Neste sentido, o presente trabalho propõe contribuições baseadas em GPUs no intuito de contribuir para o preenchimento desta lacuna na literatura. Primeiramente é proposto uma estratégia aplicadas a sistemas embarcados baseados em GPUs para identificação de sistemas não lineares e caixa-preta. Mais precisamente, foi desenvolvido uma estratégia de paralelização do algoritmo Forward Regression Orthogonal Least Squares (FROLS) para seleção de modelos autoregressivos parcimoniosos, lineares e não lineares. Em seguida são apresentadas soluções para detecção e classificação de falhas, ambas mapeadas em GPUs e baseadas na análise dos parâmetros do modelo identificado, implementadas com ferramentas de gráficos de controle (control chart) e máquinas de vetores de suporte (SVM), respectivamente. Por fim, as soluções supracitadas foram reunidas, juntamente com a estratégia de estimação recursiva dos parâmetros do modelo dentro de uma janela móvel (SWRLS), a fim de se estabelecer um algoritmo de monitoramento para detecção e diagnóstico de falhas (SWRLS_SVM_FROLS). As soluções se mostraram viáveis quando validadas com dados reais de uma pá de turbina eólica submetida a variação de temperatura e a falhas decorrentes do acúmulo de gelo e formação de trincas em diferentes escalas, apresentando a detecção e classificação correta das falhas, bem como ganhos computacionais e economia de energia, podendo chegar a 24.5x e 22x, respectivamente, para o caso de identificação/reidentificação do sistema. Vale ressaltar que este estudo de caso é atual e relevante, combinando as características de um sistema não linear variável no tempo, devido a falhas e fatores ambientais, e atuando no paradigma de computação na borda, essenciais especialmente para instalações offshore. Nesse sentido, o estudo de caso escolhido explora a solução proposta em sua totalidade, mostrando a viabilidade de um monitoramento embarcado baseado em GPU para detecção e diagnóstico de falhas em sistemas não lineares variantes no tempo.

Palavras-chave: Monitoramento de Sistemas Variantes no Tempo. Projeto de Sistemas Embarcados. Detecção e Diagnóstico de Falhas. Falhas em Turbinas Eólicas.

List of Figures

Figure 1 – Block diagram for the composition of a GPU-based embedded monitoring algorithm for fault detection and diagnosis.	34
Figure 2 – Generic potential failure curve (PF curve)	37
Figure 3 – The three classes classifier One-Against-One example with Max Win aggregation methodology.	46
Figure 4 – Interest in Edge Computing over time (data from Google Trends (GOOGLE, 2023)). The number of "search interests" is relative to the highest point on the graph for a given region in a given period. A value of 100 represents the peak popularity of a term. A value of 50 means that the term was half as popular. A score of 0 means that there was not enough data on the term.	55
Figure 5 – Comparison between CPU and GPU architecture (Figure adapted from (NVIDIA, 2023a))	57
Figure 6 – Automatic scalability of blocks in SMs (Figure adapted from (NVIDIA, 2023a)).	58
Figure 7 – Hierarchical organization of grids, blocks, and threads (Figure adapted from (KIRK; HWU, 2012)).	58
Figure 8 – Annual publications using the Jetson platform.	64
Figure 9 – Bibliometric analysis based on keywords, showing the main approaches and areas of application in works using Jetson platforms. The size of the circles is proportional to the co-occurrence of the keyword contained in it.	64
Figure 10 – Bibliometric co-citation analysis based on references applied to papers resulting from a search involving a combination of "System Identification" AND "Embedded Systems" OR "System Identification" AND "FROLS" OR "Embedded Systems" AND "FROLS".	65
Figure 11 – Bibliometric analysis based on keywords, showing the relationship between the papers resulting from a search involving the combination of the topics "System Identification" AND "Embedded Systems" OR "System Identification" AND "FROLS" OR "Embedded Systems" AND "FROLS". The size of the circles is proportional to the co-occurrence of the keyword contained therein.	66
Figure 12 – Bibliometric analysis of co-citation based on references from a search involving the combination of the topics "Fault Detection" AND "Embedded Systems".	72

Figure 13 – Bibliometric analysis based on keywords, showing the relationship between the papers resulting from a search involving the combination of the topics "Fault Detection" AND "Embedded Systems". The size of the circles is proportional to the co-occurrence of the keyword contained therein. .	73
Figure 14 – Bibliometric analysis of co-citation based on references, relating papers dealing with "Fault Detection" AND "Control Charts".	74
Figure 15 – Bibliometric analysis based on keywords, showing the relationship between the papers resulting from a search involving the combination of the topics "Fault Detection" AND "Control Chart". The size of the circles is proportional to the co-occurrence of the keyword contained therein. .	75
Figure 16 – Bibliometric analysis based on keywords, showing the relationship between the papers resulting from a search involving the topics combination of "Fault Detection/Isolation/Classification/Diagnosis AND SVM". The size of the circles is proportional to the co-occurrence of the keyword contained therein.	79
Figure 17 – Bibliometric analysis of co-citations based on the references of articles resulting from a search involving the combination of topics "Fault Detection/Isolation/Classification/Diagnosis AND SVM".	80
Figure 18 – Bibliometric analysis based on keywords, showing the relationship between the papers resulting from a search involving the combination of topics "Support Vector Machine" AND "Wind Turbine". The size of the circles is proportional to the co-occurrence of the keyword contained therein.	84
Figure 19 – Parallelization of the FROLS algorithm to the GPUs, where each block computes one of the M terms of the complete model. Note that in this case, each index of the variables g , ERR , and q represents the GPU block on which they are being calculated, and variables in bold represent vectors.	93
Figure 20 – Inner product representation between two vectors. Each thread (Th) performs the multiplication operation between the vector elements and stores the result in an internal accumulator (Acc). The elements accessed by threads are calculated according to their identifiers ID , and each thread is represented by a color.	94
Figure 21 – Representation of the parallel reduction over the cache vector allocated in the shared memory (each thread is represented by a color).	94
Figure 22 – Representation of host to device-input, device to host-output, and GPU computation time for different numbers of selected M_p terms in the parsimonious model.	97
Figure 23 – Force (input) and position (output) data of Electro-Mechanical Positioning System (EMPS).	98

Figure 24 – Evaluation of the best models obtained in identifying EMPS data when the order is varied from $nu = ny = 4$ to $nu = ny = 13$	99
Figure 25 – Results of the order model $nu = ny = 7, l = 3$ and $M = 17$: Direct comparison between the FROLS_GPU estimation and real data output in OSA (a) and Free Run (b).	101
Figure 26 – Results of the order model $nu = ny = 7, l = 3$ and $M_p = 17$: Power information in mW of the FROLS_SERIAL (a) and FROLS_GPU (b) algorithm.	101
Figure 27 – Radar chart for <i>Average Speedup</i> (left) and <i>Average Power Improvement</i> (right) Gain (xTimes) comparison varying the model orders (nu and ny) and final model terms (M_p) for EMPS case.	102
Figure 28 – Control Chart GPU - CCGPU.	107
Figure 29 – SWRLS_FROLS_GPU monitoring algorithm representing its three operating states. The blue window represents the SWRLS and error R_2 calculation, the green window represents the parametric variation verification window (W_{cc}), and the red window represents the re-identification window (W).	109
Figure 30 – SWRLS_FROLS_GPU Monitoring Algorithm Flowchart. The states are identified by colors, with state 1, corresponding to normal operation and the detection of parametric variation (green); state 2, indicating the detection of a variation and prompting the need for re-identification (red); state 3, corresponding to waiting for completion of re-identification (yellow).	109
Figure 31 – Simulated Test 1: Time-Varying Simulated System Monitoring with emphasis on the re-identification window. Background Colors: Green - Initial Model; Yellow - Parametric variation; Red - Structural variation.	116
Figure 32 – Simulated Test 2 (Data Drift Test): The drift starts at $N = 5000$, highlighted with a red background.	117
Figure 33 – Relationship between Speedup (red –Sp), Power Improvement (yellow –PI) and Error R^2 (blue –R2).	118
Figure 34 – Error of Wind Turbine System Monitoring for Added Mass Fault.	121
Figure 35 – Error of Wind Turbine System Monitoring for Crack Fault.	121
Figure 36 – Error of Wind Turbine System Monitoring - Temperature Variation.	122
Figure 37 – Error of Wind Turbine System Monitoring - Temperature Variation and Fault.	123
Figure 38 – Fault and disturbance detection algorithm flowchart. Cl corresponds to the class most indicated by the SVM classifier within the analysis window, and $\%Cl$ indicates its percentage within that window. Old_Cl corresponds to the class the system was in before parsing the current window.	131

Figure 39 – SWRLS_SVM_FROLS Monitoring Algorithm flowchart. Parameter re-estimation module (SWRLS) and R^2 calculation are shown in blue. The re-identification module shown in red, completed by the Fault Dynamics Module (in yellow), is responsible for waiting for re-identification to finish. In green is the Disturbance and Fault Detection Module.	132
Figure 40 – Adjustment of the model to the real system using the coefficient of determination R^2 for the case of faults caused by increasing crack formation. The curve in blue shows the response of a fixed model, while the one in orange shows a model response whose parameters are recursively adjusted via SWRLS and re-identified via FROLS_GPU whenever necessary. . . .	137
Figure 41 – Adjustment of the model to the real system using the coefficient of determination R^2 for the case of temperature variation and faults caused by increasing ice accumulation. The curve in blue shows the response of a fixed model, while the one in orange shows the response of a model whose parameters are recursively adjusted via SWRLS and re-identified via FROLS_GPU whenever necessary. Re-identification occurs at $N = 3013$	137
Figure 42 – Comparison between models according to the number of support vectors.	139
Figure 43 – Comparison of the F-score obtained by each tested model.	141
Figure 44 – Comparison of the execution time obtained by each tested model.	142
Figure 45 – Performance comparison among each kernel model. The metrics analyzed are F-scores, execution times (Time), speedup, power improvement, and number of support vectors (SVs), normalized between 1 and 0, with 1 being the best performance and 0 being the worst performance.	142
Figure 46 – Comparison between models with 25 parameters as features, with and without temperature.	143
Figure 47 – FDI module performance for the case of wind turbine blade cooling and increasing ice accumulation (RABC_VT case with SVM model composed of 25 parameters and RBF kernel without using temperature).	144
Figure 48 – Performance of the FDI module for the case of growing crack formation on wind turbine blades (RFIL case with SVM model composed of 25 parameters and RBF kernel without using temperature).	144
Figure 52 – Sensors configuration on the wind turbine blade (Figure adapted from (YAOWEN et al., 2021).	177
Figure 53 – Input force measurements (blue) and the output acceleration measurements (red) by channel 7 (a_7) for Healthy State (R) at 25°C.	178

List of Tables

Table 1 – Main works involving GPU-based embedded solutions for system identification.	68
Table 2 – Main works involving embedded solutions for fault detection using a control chart.	77
Table 3 – Main approaches involving fault detection and diagnosis using different techniques apart from SVMs.	78
Table 4 – Main works involving fault classification solutions using SVM.	85
Table 5 – Simulated Test 1 –Abrupt Parametric and Structural Variation	96
Table 6 – Embedded FROLS_GPU (Jetson-Nano) x SERIAL FROLS (Intel Core i7 CPU). It can be observed that the increase in the number of terms in the final model implies a better performance of the embedded FROLS_GPU algorithm compared to the non-embedded FROLS_SERIAL. However, the same does not happen with the sample increase, which the communication overhead between GPU and CPU can explain.	96
Table 7 – Comparison of GPU processing percentage in the FROLS_GPU algorithm with the variation in the number of samples N	98
Table 8 – Simulated Test 1 - Abrupt Parametric and Structural Variation: Parametric variations occur in samples $N = 3250$ ($a_1 = -1.51$ to $a_1 = 1.21$), $N = 4850$ ($b_1 = 0.6$ to $b_1 = 0.87$), $N = 6500$ ($b_2 = 0.16$ to $b_2 = 0.32$) and $N = 11380$ ($a_2 = 0.55$ to $a_2 = 0.66$).	113
Table 9 – Control Chart GPU evaluating speedup and power improvement when comparing the version running on CPU with the version on GPU, varying the size of the analysis window (W_{cc}) and the number of parameters (M_p)	120
Table 10 – 24 tested SVM models varying in kernel types and number of features. . .	129
Table 11 – Comparison between SVM models in terms of percentage accuracy (Acc) and number of support vectors (SVs).	138
Table 12 – Comparison between model complexity with Kernel RBF and Polynomials.	139
Table 13 – Models tested for the RABC_VT case.	141
Table 14 – Table with cases of failures present in the benchmark of wind turbine blades (YAOWEN et al., 2021).	178

List of abbreviations and acronyms

1DCNN	One-Dimensional Convolutional Neural Network.....	83
AGA	Adaptive Genetic Algorithm	81
AHU	Air Handling Unit.....	30
ANN	Artificial Neural Networks	41
AOA	One-Against-One	47
APMT-MODE	Adaptive Parameter control with Mutant Tournament Multi-Objective Differential Evolution).....	46
AR	Autoregressive	22
ARX	Autoregressive with eXogenous inputs.....	22
ASICs	Application Specific Integrated Circuits.....	54
AUC	Area Under the ROC Curve.....	138
CCGPU	Control Chart on GPU	44
CMSS	Common Model Structure Selection	70
CNN	Convolutional Neural Network.....	63
CPU	Central Processing Unit.....	26
CSO	Cuckoo Search Optimization	83
CUDA	Compute Unified Device Architecture	57
CUSUM	Cumulative Sum	29
CVA	Canonical Variable Analysis.....	41
DBAI	Dynamic Bayesian Anomaly Index	76
DSP	Digital Signal Processor.....	54
EMPS	Electro-Mechanical Positioning System.....	32
ERR	Error Reduction Ratio	27
ESR	Error to Signal Ratio	89
EWMA	Exponentially Weighted Moving Average	44
FDD	Fault Detection and Diagnosis	30
FDI	Fault Detection and Isolation	29

FDWT	Forward Discrete Wavelet Transform	63
FPGA	Field-Programmable Gate Array.....	54
FR	Free Run.....	100
FROLS	Forward Regression Orthogonal Least Squares	27
GLM	Generalized Linear Model.....	67
GPGPU	General Purpose Graphics Processing Unit.....	26
GPU	Graphics Processing Unit	25
GWEC	Global Wind Energy Council.....	175
HPO	Hyperparameter Optimization	46
ICA	Independent Component Analysis.....	41
IFAC	International Federation of Automatic Control.....	37
ILP	Instruction-Level Parallelism.....	56
IoT	Internet of Things	24
JIT	Just-in-Time	58
KNN	K-Nearest Neighbors.....	82
LCL	Lower Control Limit.....	42
LS-SVM	Least Squares Support Vector Machine	81
MA	Moving Average.....	44
MLP	Multi-Layer-Perceptron.....	45
MOOP	Multi-objective Optimization Problem.....	46
MPSoC	Multiprocessor Systems on a Chip.....	26
NARMAX	Nonlinear Autoregressive Moving Average with eXogenous inputs.....	22
NARX	Nonlinear Autoregressive with eXogenous inputs	22
NCA	Neighborhood Component Analysis.....	30
NIOTS	Nature Inspired Optimization Tools for SVM.....	30
NN-ESO	extended state observer neural network.....	77
NOC	Network Processors on Chip	56
NRE	Non-Recurring Engineering.....	26
O&M	Operation and Maintenance	25
OLS	Orthogonal Least Squares.....	50

OSA	One Step Ahead	100
OWT	Offshore Wind Turbine	25
PCA	Principal Component Analysis	30
PF	Potential Failure	36
PLS	Partial Least Squares	41
PSD	Positive Semi-definite	45
PSO	Particle Swarm Optimization	81
QSVM	Quadratic Support Vector Machine	82
RBF	Radial Basis Function	31
RLS	Recursive Least Squares	23
ROC	Receiver Operating Characteristic	138
SAFEPROCESS	Symposium on Fault Detection Supervision and Safety in Technical Processes	37
SDC	Silent Data Corruption	63
SEUs	Single Event Disturbances	63
SHM	structural health monitoring	176
SIMD	Single Instruction Multiple Data	57
SISO	Single-Input and Single-Output	113
SM	Streaming Multiprocessor	57
SP	Streaming processor	57
SPC	Statistical Process Control	29
SSGLR	Sequential Sampling Generalized Likelihood Ratio	76
SVM	Support Vector Machine	28
SVs	Support Vectors	30
SWRLS	Slide Window Recursive Least Squares	31
TV-NARX	NARX Time-Varying	70
UCL	Upper Control Limit	42
WMA	Wavelet Multiresolution Analysis	67
WT	Wind Turbine	25

Contents

1	INTRODUCTION	21
1.1	Contextualization	21
1.2	Problem Definition and Motivations	26
1.3	Hypotheses	27
1.4	Objectives	30
1.4.1	General Objective	30
1.4.2	Specific objectives	30
1.5	Methodological Aspects	31
1.6	Contributions	32
1.7	Document Organization	35
2	THEORETICAL BACKGROUND	36
2.1	Fault Detection and Diagnosis	36
2.1.1	Nomenclature	37
2.1.2	Approaches to Fault Detection and Diagnosis	39
2.1.3	Residue Analysis	41
2.1.4	Control Charts	42
2.1.5	SVM Classifiers	44
2.2	Systems Identification	47
2.2.1	NARMAX/NARX Polynomial Model	48
2.2.2	FROLS strategy based on orthogonal descomposition	50
2.2.3	FROLS strategy to get the best terms	52
2.2.4	Adjusting model parameters via Slide Window Recursive Least Squares .	53
2.3	GPU-Based Embedded Systems	54
2.3.1	Graphics Processing Units - GPU	56
2.3.2	Jetson Nano	59
2.4	Final Chapter Remarks	59
3	RELATED WORKS	61
3.1	GPU-based Embedded Platforms - Jetson	62
3.2	GPU-Based Embedded Solution for System Identification	65
3.3	Embedded-Based Solutions for Monitoring and Fault Detection . .	69
3.4	Fault Classification Based on SVM and NARX Model	77
3.5	Final Chapter Remarks	85

4	GPU-BASED APPROACH FOR EMBEDDED NARX/FROLS SYSTEM IDENTIFICATION	87
4.1	Proposed Methodology	88
4.1.1	The overall FROLS algorithm	88
4.2	The strategy of mapping FROLS to GPU	90
4.2.1	The GPU platform used and the overall strategy	91
4.2.2	Defining a parallelization strategy through a theoretical model	91
4.2.3	Validation of the parallelization strategy	93
4.3	Results	94
4.3.1	Performance Analysis using a simulated theoretical model	95
4.3.2	Performance Analysis - EMPS Case Study	97
4.4	Discussion	102
4.4.1	About the GPU approach	102
4.4.2	About results	103
4.5	Final Chapter Remarks	104
5	GPU-BASED EMBEDDED MONITORING FOR FAULT DETECTION IN TIME-VARYING NONLINEAR SYSTEMS	105
5.1	Proposed Methodology	106
5.1.1	Control Chart GPU (CCGPU)	106
5.1.2	The SWRLS_FROLS_GPU	108
5.1.3	Set of Tests and Performance Analysis	111
5.2	Results	115
5.2.1	Time-Varying Simulation Model Results	115
5.2.2	Wind Turbine Blade Results - The Case Study	118
5.3	Discussion	123
5.4	Final Chapter Remarks	124
6	EMBEDDED SVM-BASED CLASSIFIER FOR FAULT DETECTION AND DIAGNOSIS	126
6.1	Proposed Methodology	127
6.1.1	SVM-Based Disturbance and Fault Detection Module	128
6.1.2	SWRLS_SVM_FROLS Monitoring Algorithm	131
6.1.3	Set of Tests and Performance Analysis	134
6.2	Results	136
6.2.1	Identification/Re-Identification Module - FROLS_GPU	136
6.2.2	SVM_GPU evaluating performance and power consumption	138
6.2.3	Monitoring Valuation	139
6.3	Discussion	145
6.4	Final Chapter Remarks	147

7	CONCLUSIONS	148
7.1	Future Works	149
	REFERENCES	151
	APPENDIX	168
	APPENDIX A – RESUMO ESTENDIDO EM LÍNGUA PORTUGUESA	169
	APPENDIX B – CASE STUDIES	175
B.1	Wind Turbines - A Case Study	175
B.1.1	Benchmark Used	176
B.2	Electro-Mechanical Positioning System (EMPS)	177
	ANNEX	180
	ANNEX A – ROC CURVE OF THE SVM_GPU CLASSIFIER	181

1 Introduction

This chapter addresses the general aspects on which this thesis was developed, highlighting the contextualization, motivation, and hypotheses that underlie it, as well as the established objectives and proposed contributions.

1.1 Contextualization

Understanding real-world systems, whether physical, biological, or financial systems, is essential to predicting their functioning, controlling, simulating, and even optimizing them to meet specific objectives. The engineering field exemplifies this requirement, as there is a rising call for food, energy, transportation, services, and general products, driving progress in development and production methods to optimize the efficacy of these systems (LUCKE, 2022).

Industrial systems are an excellent illustration of this transition due to their significant role and evolution. They have evolved from the use of only mechanical equipment to including electricity, automation, artificial intelligence, and robotics. This has increased their efficiency and complexity (PIVOTO et al., 2021). In this context, there is significant worry regarding the quality, safety, reliability, and accessibility of production systems and their equipment. These factors ensure the protection of operators and consumers and influence their efficiency, leading to financial implications. Therefore, understanding these systems becomes even more pertinent (DE SOUZA et al., 2022).

Despite the recent advancements in sensing technology and component quality in engineering systems, it is impossible to ensure fault-free operations. As a result, the imperative of process monitoring and supervision, commonly referred to as *condition monitoring*, becomes manifest for the assurance of safety, reliability, and system availability (DE SOUZA et al., 2022). It is worth emphasizing that substantial investments are customarily directed toward maintenance endeavors to realize the objectives mentioned above. Such investments enhance overall efficiency and lead to shorter payback periods. Furthermore, these proactive measures directly diminish the incidence of faults and facilitate their early detection, thereby resulting in reduced repair costs and an overall enhancement of system safety (CHEN, L. et al., 2023).

The presence of a wide range of factors underscores the substantial commercial and scientific significance of the task of monitoring systems. Consequently, this area of study is expanding in scope, owing to recent technological advancements and innovative tools that provide fresh insights and solutions to address this challenge. The task of monitoring

extends beyond the industrial sector to cover various areas, including robotics (NATARAJAN et al., 2023; OH et al., 2022), aviation (WANG, H. et al., 2019), automation (KHANNA et al., 2023), health (SHEN, M. et al., 2022), agriculture (WONGCHAI et al., 2022), economics, and biology, among others.

The system's complexity under observation and its operational conditions play a pivotal role in determining the level of challenge inherent in the monitoring task, particularly regarding the quality of fault detection and investigation. Nevertheless, the growing significance and interest in this subject matter have given rise to numerous research endeavors dedicated to developing effective methodologies for investigating faults in dynamic systems. These research works exhibit variations in depth and chosen methods (see Chapter 2). In this context, two main approaches are employed to develop monitoring systems for detecting and analyzing faults: model-based and data-driven approaches.

A fundamental challenge inherent in the model-based approach resides in the accuracy issue. This challenge stems from the requisite physical knowledge needed to construct an analytical mathematical model of the monitored system may not always be accessible, rendering an equivalent representation of the actual system difficult or even unattainable. In response to this limitation, system identification techniques are extensively employed to establish a model based solely on a system's input and output data. This can be accomplished through methods such as black-box identification or gray-box identification when partial knowledge of the system's physical principles is available. Such an approach aligns with the data-driven methodology, allowing for the empirical acquisition of a model (SILVIO SIMANI, 2018).

A sound data-driven approach for monitoring and detecting faults comprises two fundamental elements. The first involves the acquisition of a model representing the monitored system. In contrast, the second entails the identification and diagnosis of faults through the observation of disparities between the behavior of the actual system and the established model. Several techniques are available for obtaining these models using data from the relevant system. These techniques encompass a wide range, from mathematical series to machine learning techniques. Examples of parametric models in time series analysis include *AutoRegressive* models with or without exogenous inputs, namely *autoregressive* (AR), *AutoRegressive with eXogenous inputs* (ARX), *Nonlinear AutoRegressive with eXogenous inputs* (NARX), and *Nonlinear AutoRegressive Moving Average with eXogenous inputs* (NARMAX) (BILLINGS, 2013). These models are widely used due to their applicability to a wide range of linear and nonlinear systems. They are also relatively easy to comprehend and implement, offering good identification performance at a reasonable computational cost (AGUIRRE, 2015).

From a monitoring perspective, the identification of systems can present some challenges. Despite an adequate initial model being in place, there is always a chance that, over

time, a divergence may occur between the identified model and the actual response of the system. This divergence occasionally occurs suddenly when abrupt faults arise, resulting in a bias in the monitored signal that is not difficult to detect. However, gradual change may occur in some cases due to incipient faults and environmental variations. This deviation in data is termed concept drift or data drift (AGRAHARI; SINGH, 2022; TSYMBAL, 2004). In practice, numerous actual systems exhibit temporal variation in their dynamics, attributed to environmental factors, including temperature, wear and tear, or even a shift in their operating mode. This results in data drift and necessitates a precise identification strategy (AGUIRRE, 2015). Data drift is a frequent issue in monitoring applications that have seen increased focus in recent years in fields such as fault detection, sensing, and robotics (ANDRÉS et al., 2023). As a result, online monitoring solutions must be engineered to accommodate continuous learning scenarios and effectively manage non-stationarities, changes, and deviations in data (ELWELL; POLIKAR, 2011).

Considering the information presented earlier, a fixed model proves inadequate for addressing data drift regardless of the fluctuation's origins in system dynamics. Consequently, it becomes imperative to introduce modifications responsive to these changes. In this context, the utilization of parametric models may lead to subtle variations within the system, resulting in a disparity between the system's output and the employed model. The process of recursive parameter estimation, such as *Recursive Least Squares* (RLS) algorithm, offers a means to rectify these variations. This is recognized as a parametric adaptation of the system. However, there are instances when the system's dynamics undergo significant alterations, rendering it unfeasible to reconcile its output with the model's output solely through parameter adjustments. In such cases, a new system identification is necessary to establish a novel model with a distinct structure. An occurrence of this nature signifies a structural modification in the system. Many real-world systems exhibit parameter and structural variations and are categorized as parameter and structure variant systems. (BILLINGS, 2013). To summarize, whether the variations are due to changes in parameters, structure, or both, a monitoring system must provide an adaptive identification of the monitored system, which is typically done online. This task can be challenging and may require a substantial computational cost.

Once system identification has been completed, obtaining a model that represents the interest characteristics of the monitored system with the intended accuracy, the next step is to create a strategy to verify any variations between the model and the real system, which could potentially indicate faults. It is crucial to detect and pinpoint the source of these variations. One significant concern within the domain of monitoring solutions is the ability to distinguish between the various phenomena responsible for data drift. This capability is pivotal for making informed decisions that uphold the standard operation of the monitored system, preventing unwarranted actions or omissions in response to genuine requirements. More precisely, monitoring assumes a pivotal role in the early detection of potential system

faults whenever feasible. This underscores the importance of identifying incipient faults that manifest gradually, rendering their detection challenging and potentially causing the monitoring algorithm to overlook them.

It is evident that identifying an incipient fault in the early stages is advantageous, as it allows for preventative measures to be taken to prevent further damage to the system and any potential failure, thereby affecting security and necessitating corrective maintenance, which typically incurs higher financial costs (CHEN, L. et al., 2023). From the above, it is evident that monitoring systems with varying dynamics pose inherent challenges, whether due to the necessity for adaptive system identification or the task of detecting and diagnosing faults, particularly in the instance of incipient faults during their initial stages. This challenge assumes even greater significance when one considers the deployment of such solutions in embedded systems and the accompanying constraints associated with them. Despite the challenges, the development of applications based on the *Internet of Things* (IoT) (KAUR et al., 2023) and *Edge Computing* (SHI et al., 2016; DAI et al., 2019) has led to increasing interest and potential for creating dedicated monitoring systems embedded in physical systems. The aim is to bring data processing closer to its source. Dedicated systems offer significant benefits for monitoring various areas of interest. For instance, they enhance information processing speed by reducing the distance between data processing and its source. Additionally, such solutions facilitate immediate data analysis and minimize data flow, enhancing the security of receiving and processing data. Furthermore, an economic advantage is present as embedded systems usually do not require substantial capital investments (MARWEDEL, 2010). Additionally, this approach can aid in implementing monitoring systems in remote environments, such as offshore systems.

A contemporary and pertinent illustration of systems that conform to the patterns mentioned earlier, subject to fluctuations in their dynamics and with offshore installations, is *Wind Turbines* (WTs). These have become prominent due to their impartiality to the environment and favorable cost-effectiveness ratio within a scenario of apprehension regarding the production of energy in a hygienic, sustainable, and effective manner (SILVIO SIMANI, 2018; ARCOS JIMÉNEZ et al., 2020). Many wind farms are located offshore due to access to higher quality wind energy, the potential of installing larger turbines, and the avoidance of operational issues such as noise, visual pollution, and other disturbances to daily life (SILVIO SIMANI, 2018). Nevertheless, this hostile operating environment imposes greater vulnerability on the system, increasing the risk of unexpected breakdowns and significant damage. Furthermore, restricted accessibility to these offshore structures results in increased *Operation and Maintenance* (O&M) expenses compared to those of onshore systems. According to statistics (HEVIA-KOCH; KLINGE JACOBSEN, 2019), O&M costs constitute approximately 20% to 25% of the total cost of an *Offshore Wind Turbine* (OWT). It is essential to consider that the implementation of OWTs requires an effective monitoring system to ensure higher levels of reliability, safety, and maintenance convenience compared to onshore

projects (IZQUIERDO et al., 2020; GAO, Z.; LIU, X., 2021; DIBAJ; GAO, Z.; NEJAD, 2023).

Currently, most WTs utilize a simplistic conservative protocol for fault management, involving system shutdown and waiting for maintenance. In light of this practice and offshore system characteristics, an effective monitoring solution may be found in embedded systems that prioritize the isolation of installations and align with edge computing paradigms. In this sense, a specialized system would undertake both the collection and analysis of data for detecting and diagnosing possible faults, thereby offering the various benefits these systems inherently provide (see Section 2.3).

However, conventional monitoring solutions entail substantial computational expenses, coupled with factors such as memory and energy consumption, rendering them unsuitable for use in embedded systems (MARWEDEL, 2010). A noteworthy instance of this is the employment of the system identification task, used as the first stage of a data-driven approach to fault analysis, as previously discussed. Naturally, system identification can be executed offline for some applications, mitigating this issue. However, this alternative is not feasible for applications dealing with systems subjected to variations in parameters and structure. The reason is that the solution must be able to re-identify the monitored system whenever necessary and online. Thus, it becomes essential to modify traditional system identification techniques and the entire fault monitoring and diagnosis process to accommodate the constraints of embedded systems.

Within the context presented, **there is a gap** in the literature, with few studies addressing the challenge involved in designing an embedded monitoring solution for detecting and diagnosing faults in nonlinear and time-varying systems (see Chapter 3), whether from a hardware, software or combined approach, known as "hardware/software co-design" (SHAOUT; EL-MOUSA; KHALID, 2009). There is an increasing demand for solutions that meet the requirements mentioned earlier effectively and efficiently. This field presents significant potential for search, especially considering the advancement of machine learning tools, embedded microprocessors, and development platforms, in particular, in terms of their computing power.

In this context, this work presents a method for embedded monitoring time-varying systems, utilizing *Graphics Processing Units* (GPUs) for fault detection and diagnosis. The strategies for identifying systems and detecting and diagnosing faults are described in detail, alongside an analysis of computational costs and energy consumption. The proposed solution was tested for feasibility by inputting it into a database that takes into account environmental factors (temperature) and 12 types of faults found in wind turbine blades (which includes faults of ice accumulation and crack formation - see Appendix B). It is important to note that these blades are critical components of wind turbines, and any failure in these elements has significant consequences, leading to massive losses, especially in finance. Moreover, identifying such faults presents a greater challenge than detecting other potential issues that

may impact wind turbine systems (CIANG; LEE; BANG, 2008).

1.2 Problem Definition and Motivations

As expounded in Section 1.1, the incorporation of monitoring systems is paramount across diverse engineering domains, primarily about the facets of fault detection and diagnosis. Real-world systems invariably exhibit dynamic fluctuations, which may stem from environmental influences, the effects of wear and tear, disparities in operational modes, and the vulnerability to various forms of disturbances and noise. All these factors contribute to the challenges of monitoring tasks, including correctly identifying the mathematical model representing the system, enabling the model to adjust to faults, and distinguishing faults from other factors that may cause data drift.

In this sense, a good monitoring solution must meet the requirements highlighted above, adapting to the scenario of continuous learning to maintain the quality of the system model online and perform fault detection and diagnosis correctly. Thus, monitoring time-varying systems for fault detection and diagnosis is a complex task, and its challenge is even more significant when applied to embedded systems and their constraints.

Given this difficulty, few studies have addressed this task, and there is a gap in the literature regarding viable embedded solutions. Therefore, this work proposes a GPU-based embedded solution for monitoring time-varying systems to detect and diagnose faults, aiming to bridge this gap. The use of general purpose GPUs (GPGPUs) has risen recently due to the emergence of heterogeneous embedded platforms (*Multiprocessor Systems on a Chip* - MPSoCs). These platforms combine *Central Processing Units* (CPUs) with multicore architectures, like GPUs, primarily for edge computing, IoT, and artificial intelligence applications. Thus, these components are feasible to be applied to edge processing systems, including the monitoring task, by utilizing GPUs' parallel processing power combined with CPUs' computational capabilities. This, in turn, facilitates the creation of high-performance embedded solutions. Furthermore, GPUs offer a favorable cost-benefit ratio due to their good energy and computational efficiency, complemented by low *non-recurring engineering* (NRE) expenses.

Few studies on this subject offer a detailed analysis of the proposed strategy's computational capacity or energy consumption concerning the problem's dimensionality. These are crucial factors within embedded systems. This work addresses these aspects, laying out the proposed strategies and comparing configurations to guide solution capacity under such settings. As this solution can apply to various physical systems with distinct dimensions and characteristics, it is crucial to make this comparison in the preliminary stages to determine the feasibility of the solution for the problem under analysis.

The proposed solution has undergone validation via a current, relevant benchmark,

utilizing a data set that contains fault information and temperature variation in a wind turbine blade (see Appendix B). Currently, this area has considerable commercial and academic interest, notably in solutions for use in offshore installations, which provides a favorable scenario for implementing embedded monitoring systems. It is worth mentioning that, although the solution presented in this study has been validated for monitoring wind turbines, it is also flexible enough to be applied to linear and non-linear systems that generally vary over time. Additionally, this work facilitates the development of future solutions in monitoring and fault detection. For example, one potential application is creating an IoT network for monitoring wind farms and fault-tolerant control applications.

1.3 Hypotheses

- *First hypothesis:* Given the efficiency of simplified NARX models via the *Forward Regression Orthogonal Least Squares* (FROLS) algorithm for nonlinear black-box systems, despite the sequential nature of this algorithm, if the calculation of the terms contribution by the *Error Reduction Rate* (ERR) criterion can be parallelized via GPUs, then it will be possible to generate computational and energy savings without losing the quality of the identification. This will favor the use of the FROLS algorithm in embedded systems.
- *Second hypothesis:* Given the fact that a parametric model has been used to identify the monitored system, if a GPU-based control chart tool is applied to the system model parameters, together with the residual analysis, then there could be an improvement in the performance of fault detection solutions, especially for the case of incipient faults which, especially in their early stages, can be masked by the recursive parametric adjustment of the model. In addition, mapping on GPUs would bring improvements in computational time and energy consumption, essential factors in the context of embedded systems.
- *Third hypothesis:* Still considering the fact that a parametric model is used to identify the monitored system, if a GPU-based *Support Vector Machine* (SVM) classifier is built in which its hyperparameters are optimized and the parameters of the regressive model obtained in the identification are used as input features of the SVM model, then an efficient solution can be obtained for detecting and classifying (isolating) faults in time-varying systems. It is worth remembering that the terms of the identified autoregressive model were chosen according to a criterion of relevance based on ERR. Thus, part of this hypothesis is that it is not necessary to use all the parameters of this model as SVM features, but only the first ones, since they accompany the most relevant terms, reducing the amount of data to be processed. It is therefore hoped that the efficiency of the SVM classifier will be achieved from the point of view of classification

assertiveness and a computational and energy point of view. These characteristics would result from having a simpler SVM model in terms of support vector number, optimizing the hyperparameters, reducing the number of features, and implementing the SVM kernel on GPUs.

Concerning the first hypothesis, its importance lies in that system identification is one of the pillars of data-driven system monitoring for fault detection and diagnosis. In the case of time-varying systems, it is not enough to have a fixed model, and a solution is needed that can re-identify the system whenever there is a change in its operating dynamics. However, conventional identification techniques generally require high computational costs and energy consumption, which can be limiting factors when considering applications using embedded systems. One of the most widespread approaches is identification using autoregressive parametric models, such as NARX and NARMAX, which use the FROLS algorithm to select the simplified model, proving to be effective for a wide range of applications (BILLINGS, 2013; LI, Y. et al., 2020a; LI H.-L. W., S. A. B., 2016). In this sense, the method for calculating the terms contribution of the simplified auto-regressive model using the ERR criterion consists of iteratively calculating the contribution of each of the original model terms, which, depending on the problem, can run into thousands (see subsection 2.2.2). Therefore, if the hypothesis is true, the computational and energy gains are expected to increase with the complexity of the problem, since a greater number of terms would be needed in the simplified model, further favoring the improvement obtained with parallelization on GPUs. Research in the literature has shown that there are no studies that develop a detailed analysis of the parallelization of the FROLS algorithm, presenting aspects of the strategy used and time and energy performance for embedded systems, primarily using GPUs (see section 3.2 of Chapter 3). For example, the work by (BEHARRELL; HONARY, 2016) was the closest to the subject, citing a parallel version of an algorithm similar to FROLS on GPUs. However, there are no details on this implementation. The authors do not discuss aspects of parallelization strategies and their consequences for computational efficiency and energy consumption, including the communication bottleneck between CPU and GPU, which is often a critical performance problem in this approach. It is worth noting that in addition to detailing the mapping strategy, an analysis of its performance using different configurations is of the utmost importance to gain an initial view of the proposed solution's viability. This is because the complexity of the system identified will directly affect these configurations, such as the number of terms needed in the simplified model to ensure good identification.

With regard to the second hypothesis, it is known that another fundamental pillar of system monitoring aimed at fault detection and diagnosis is the solution's ability to detect the occurrence of such faults. One of the most widely used solutions is residue-based analysis, which is based on the residues generated between the responses of the model and the real system to decide whether a fault has occurred (SIMANI; FANTUZZI; PATTON,

2003). However, it is known that this method has some weaknesses, such as its sensitivity to disturbances and noise (generating false positives) and its inability to detect faults if there are parametric adjustments to the model, as in the case of using recursive estimation algorithms (generating false negatives) (FLEISCHER et al., 2014). So, once you have a parametric model representing the system of interest, an alternative is to use the information in these parameters to help detect the occurrence of faults. This parametric analysis can be carried out in isolation or in conjunction with residual analysis to make the detection solution more reliable. In this context, the control chart, which is a *Statistical Process Control* (SPC) technique, is often used to monitor the variable outputs of dynamic systems (see section 2.1.4) (MONTGOMERY, 2009). This technique is based on monitoring the variations undergone by the system's variables to the point of classifying them as *in control* or *out of control* and can be applied to the parameters of the monitored system model to help with the task of fault detection (HELBING; RITTER, 2017). A few papers deal with this subject (see section 3.3), such as (TRAN; FOWLER, 2020), where a *fault detection and isolation* (FDI) solution is proposed for lithium battery sensors, using RLS and *cumulative sum* (CUSUM) with moving average. However, none of the papers dealt with mapping the control chart solution to GPUs and its application in embedded systems.

About the third hypothesis, it is known that, in addition to the task of fault detection, another need may be to classify the fault that has occurred according to a set of cataloged faults, a task known as fault isolation. Various approaches are used for this purpose, with SVM being a well-established approach (VAPNIK, 1995). It should be remembered that obtaining the parameters of an SVM model represents a convex problem, which guarantees the optimal solution within the specified constraint space. However, the quality of SVM classifier models depends on factors such as the selection of features, the kernel, and the hyperparameters used, directly implying their complexity, which can be a crucial factor for their application in embedded systems (SILVA SANTOS et al., 2021). An important aspect of using SVMs concerns the selection of the features used, with algorithms such as *principal component analysis* (PCA) and *Neighborhood Component Analysis* (NCA) commonly being used. However, in the system identification stage, the terms that make up the final model were chosen in order of relevance according to the ERR criterion. Thus, part of the hypothesis described above is that it would not be necessary to use all the parameters of the simplified model since the quality of the classification could be maintained by using a reduced number of parameters starting from the first terms. Thus, the hypothesis is based on the assumption that optimizing the SVM model's hyperparameters via *Nature Inspired Optimization Tools for SVM* (NIOTS) (see section 2.1.5), mapping the kernel to the GPU and using the first parameters of the monitored system's model as features would generate an accurate and simplified SVM classifier, in terms of *Support Vectors* (SVs), thus helping to reduce processing time and energy consumption, which are essential for its use in embedded systems. Few studies use parameters from models like NARX as features in SVM classifiers (see section

3.4), such as (AFSHARI et al., 2015), which presents a hybrid method combining ARX and SVM for *Fault Detection and Diagnosis* (FDD) applied to an *Air Handling Unit* (AHU) problem.

Once the above hypotheses are confirmed, it will be possible to establish a monitoring algorithm for fault detection and diagnosis of nonlinear time-varying systems. Fault detection can be carried out either as described in the second hypothesis or using the SVM classifier, which also acts to isolate the fault, as described in the third hypothesis. Thus, once detected and isolated, the system dynamics under the fault condition can be captured through online re-identification, as described in the first hypothesis, thus closing the fault diagnosis stage.

1.4 Objectives

The objectives of this work are divided into general objectives, which present the main focus of this study, and specific objectives, which are the results to be achieved in the process of reaching the general objective.

1.4.1 General Objective

The general objective of this thesis is to develop tools that enable an embedded GPU-based solution for monitoring nonlinear time-varying systems for fault detection and diagnosis, presenting a detailed analysis of the strategies adopted, as well as the metrics of computational time and energy consumption, which are fundamental for the context of embedded systems.

1.4.2 Specific objectives

The specific objectives of this work are:

- (a) To propose a strategy for parallelizing the FROLS algorithm for GPUs in order to have a solution that allows the identification of systems via polynomial autoregressive models with reduced computational time and energy consumption.
- (b) Carry out an analysis of the computational and energy performance of the FROLS algorithm mapped to GPUs in different configurations of its hyperparameters.
- (c) Developing a strategy based on control charts and mapped on GPUs for analyzing parameter variations in a NARX model in order to detect the occurrence of faults in the monitored system.
- (d) Develop a monitoring algorithm for fault detection based on SWRLS, parametric variation and residue analysis, capable of re-identifying the monitored system online.

- (e) Develop an SVM model with hyperparameter optimization and using the parameters of a NARX polynomial system model as features, in order to classify types of faults in dynamic systems.
- (f) Carry out an analysis of the SVM classifier's accuracy, using conventional metrics from the literature, as well as analyzing computational and energy performance for polynomial and *Radial Basis Function* RBF kernels (both mapped on GPUs) and by varying the number of features used.
- (f) Develop a monitoring solution for fault detection and diagnosis using a GPU-based SVM classifier, SWRLS and online re-identification of the system whenever necessary.
- (g) Validate the proposed solutions with benchmarks from a real system containing data from wind turbine blades subjected to temperature variation and failures due to ice build-up and crack formation, available in the literature.

1.5 Methodological Aspects

To achieve the objectives mentioned in section 1.4, the following methodological procedures were adopted:

- (a) Analysis of possible strategies for mapping the FROLS algorithm to GPUs by performing a profile of the algorithm in order to identify the different computational costs and parallelization potential of its component parts.
- (b) Mapping the FROLS algorithm to GPUs, called FROLS_GPU.
- (c) Validation of the FROLS_GPU solution in a real case study (Section B.2 of Appendix B), analyzing the quality of the identification and the computational and energy performance for different tunings of its hyperparameters.
- (d) Analysis of possible mapping strategies on GPUs for a control chart tool based on CUSUM and moving average, called CCGPU, in order to detect system failures by varying the parameters of its model.
- (e) Implementation of the CCGPU tool and analysis of its computational and energy performance.
- (f) Development of a monitoring algorithm for fault detection using SWRLS, CCGPU and FROLS_GPU, called SWRLS_FROLS_GPU.
- (g) Validation of the SWRLS_FROLS_GPU algorithm in a real wind turbine blade case study (Section B.1 of Appendix B).

- (h) Optimization of the hyperparameters of a GPU-based SVM classifier (SVM_GPU) via NIOTS, as well as the mapping of polynomial and RBF kernels to GPUs.
- (i) Validation of different SVM_GPU classifiers in a real case study (Section B.1 of Appendix B), differing in the type of kernel (polynomial and RBF) and the number of parameters of a NARX polynomial model used as classifier features, with and without temperature information.
- (j) Performance analysis of the SVM_GPU classifier from the point of view of accuracy, computational time and energy consumption.
- (l) Development of a monitoring algorithm for fault detection and diagnosis, called SWRLS_SVM_GPU, based on SWRLS, SVM_GPU and FROLS_GPU, capable of identifying faults, classifying them and capturing the new system dynamics resulting from their occurrence (fault detection and diagnosis).
- (m) Validation of the SWRLS_SVM_GPU algorithm in a real wind turbine blade case study (Section B.1 of Appendix B).

1.6 Contributions

Based on the literature critical review (presented in Chapter 3), and in order to help fill the gaps found, the following contributions have been proposed:

GPU-based Approach for Embedded NARX/FROLS System Identification (Presented in Chapter 4)

Since there is a gap in the literature regarding the application of system identification solutions to embedded systems, this work proposes:

- A novel version of the FROLS algorithm addressed to GPU (the FROLS_GPUS), which was focused on computational performance and energy efficiency.
- An analysis of the proposed solution exploring aspects of computational performance and energy consumption by varying the dimensionality of the problem (reflected in the complexity of the model that represents it).

The proposed solution was validated with a real nonlinear *Electro-Mechanical Positioning System* (EMPS) benchmark related to the friction commonly found in robotic manipulators.

This chapter has been submitted for review: M. M. Soudre, H. V. H. Ayala, A. C. Melo, C. H. Llanos, A novel GPU-based approach for embedded NARMAX/FROLS system identification. Under Review, 2023.

GPU-Based Embedded Monitoring for Fault Detection in Time-Varying Nonlinear Systems (Presented in Chapter 5)

To address the lack of available literature on monitoring solutions for fault detection applied to embedded systems, this work proposes:

- Construction of a continuous learning algorithm for monitoring and fault detection of systems based on parameter analysis via SWRLS and control chart, named SWRLS_FROLS_GPU. The algorithm can identify parametric variations (treated with SWRLS) and structural variations (re-identifying the system online via FROLS mapped on a GPU-based platform).
- A control chart strategy designed to be implemented on a GPU-based embedded system, as well as an analysis of its computing capacity and energy consumption. To the best of the authors' knowledge and based on extensive research, no existing literature has documented the mapping of control chart algorithms for GPUs in the context of online failure monitoring. Results showed computational gains surpassed 10x, and energy consumption improved by almost 30x when comparing the CCGPU variation detection module with the serial version of the same algorithm.

It is important to highlight that the proposed solution was validated with a data benchmark of a WT system with the occurrence of abrupt, incipient faults and environmental temperature variations. The solution accurately detects variations experienced by the system, whether resulting from temperature changes or the onset of sudden or incipient failures.

This chapter has been submitted for review:

M. M. Soudre, H. V. H. Ayala, C. H. Llanos, GPU-Based Embedded Monitoring for Fault Detection in Time-Varying Nonlinear Systems - A Wind Turbine Case Study. Under Review, 2023.

Embedded SVM-based Classifier for Fault Detection and Diagnosis (Presented in Chapter 6)

Still regarding the time-varying systems monitoring, and in order to help fill the existing gap in the literature regarding solutions not only for detection, but also for dynamic fault detection using embedded systems, this work proposes:

- GPU-based embedded algorithm for monitoring and FDD of linear and nonlinear time-varying systems, dealing with parametric and structural variations, called SWRLS_FROLS_SVM. The algorithm uses SWRLS to adjust parameters and FROLS_GPU to re-identify

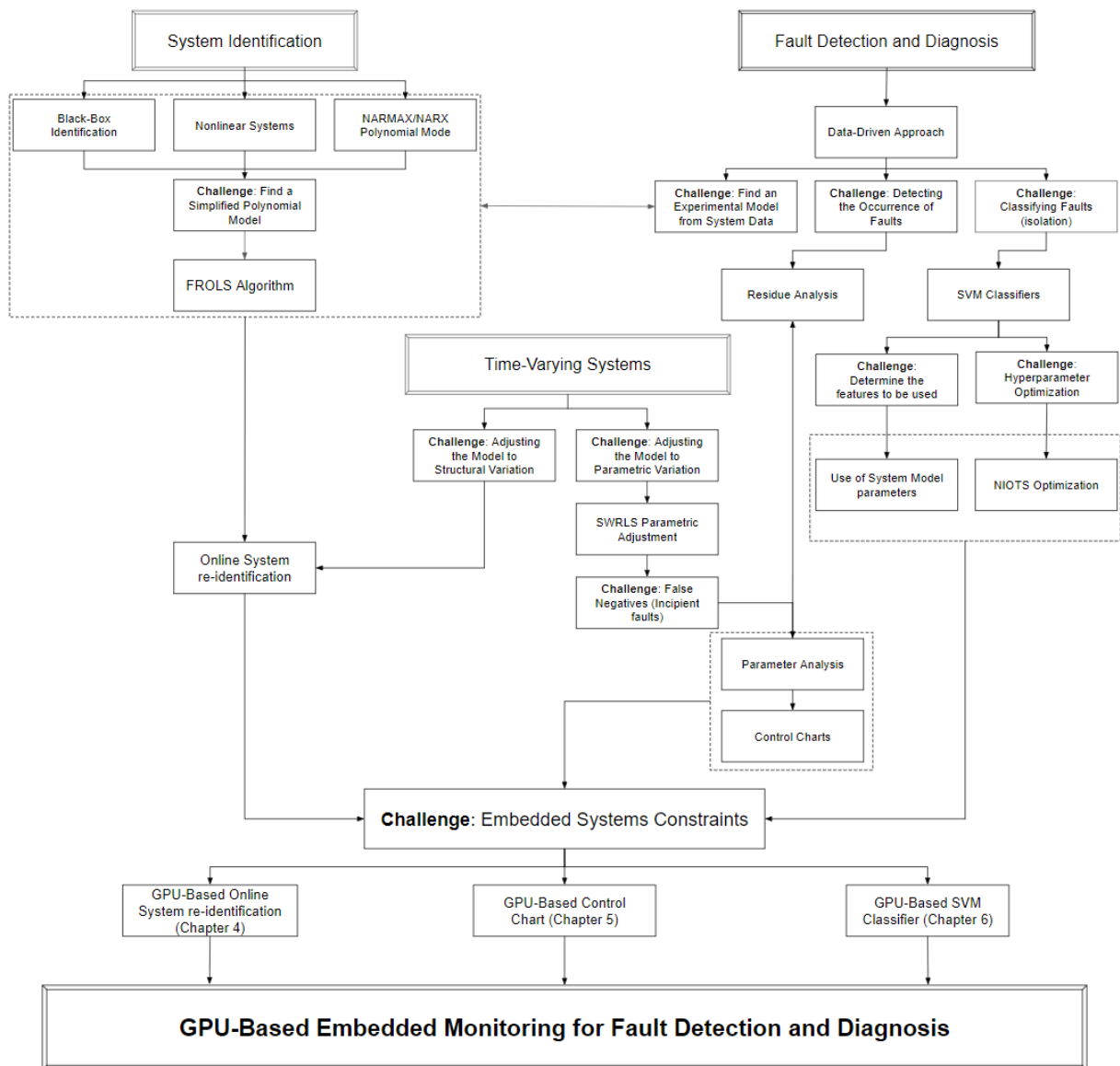


Figure 1 – Block diagram for the composition of a GPU-based embedded monitoring algorithm for fault detection and diagnosis.

the system online when necessary. Fault detection and isolation are done via an optimized SVM model mapped onto GPUs whose features are the parameters of the system's NARX model. Once the fault has been detected and the variation in system dynamics resulting from it has been verified, re-identification takes place, thus obtaining a new model that represents the system's dynamics after the fault has occurred, completing the FDD solution.

- A module for detecting faults and disturbances in dynamic systems applied to embedded systems, comprising a GPU-based SVM classifier with features corresponding to the parameters of a NARX model. The optimizations of the SVM classifier include the use of NIOTS (SILVA SANTOS et al., 2021), generating the optimized hyperpa-

rameters, and choosing the number of parameters of the NARX model to be used. This optimization aims to (a) reduce the complexity of the model (number of SVs), contributing to savings in memory, execution time, and power consumption of the embedded system; (b) increase the accuracy of the model, giving reliability to the fault detection and diagnosis process. The results showed good performance in terms of fault detection and classification accuracy, as well as the feasibility of implementation in embedded systems, with savings in execution time and energy consumption of up to 3.5x when compared to the serial version of the solution.

- Detailed discussion of the cost-benefit of the above solution when using polynomial and RBF kernels, both mapped on GPU, in terms of execution time, energy consumption, SVM model complexity, and fault classification efficiency.

The proposed solution was validated with a benchmark of WTs that combine faults (ice accumulation and cracks in the blades) and temperature variations. The results showed the solution's viability and its accuracy in detecting faults and disturbances caused by environmental variation. This chapter is being prepared for submission.

Figure 1 illustrates the proposed contributions through a block diagram highlighting the approaches used and the challenges involved.

1.7 Document Organization

The remaining chapters of this document are divided as follows: Chapter 2 presents the theoretical background, where the concepts of fault detection and diagnosis, system identification, and GPU-based embedded systems are described. Chapter 3 presents a critical review of the literature, showing the gaps in the area under discussion, as well as the contributions proposed to fill these gaps. In Chapters 4, 5, and 6, details of the proposed contributions are presented. In this sense, Chapter 4 presents the embedded solution based on GPUs for system identification via FROLS algorithms. Chapter 5 presents the monitoring solution for detecting faults in a wind turbine system, using parametric analysis via a control chart mapped onto GPUs. In Chapter 6, the solution for detecting and diagnosing faults is presented using an optimized SVM model with features from the monitored system's NARX model parameters. Finally, Chapter 7 presents the general conclusions of the work, as well as a discussion of possible future work.

2 Theoretical Background

This chapter deals with the theoretical basis on which this work was developed. In this sense, the following sections will present details involving fault detection and diagnosis (2.1), system identification (2.2), and GPU-based embedded systems (2.3).

2.1 Fault Detection and Diagnosis

In the previous chapter, the general aspects of monitoring dynamic systems were discussed, as well as their importance, especially in fault detection and diagnosis. Technological advances, market competition, and increased demand for products, services, and energy have driven the need for physical systems' efficiency, especially regarding factors such as safety, durability, availability, and reliability (DE SOUZA et al., 2022). In addition to these factors, the high complexity of such systems leads to higher costs, which further justifies concern about their efficiency to minimize expenditure on operation and maintenance services (CHEN, L. et al., 2023). Within this context, fault detection is fundamental both from the point of view of safety, functionality, and efficiency of physical systems and from a financial point of view since detecting faults in their early stages or even preventing them will have a direct positive impact on all these points.

In this sense, the objective is to formulate maintenance plans focusing on reliability. These plans will utilize fault detection tools to predict or prevent faults, detect them early, and minimize corrective actions (CARRIR; NUNES; TERRA, 2020). An example of this is the *potential-failure* curves (PF curves), which were initially published in a study supported by United Airlines in 1978 (NOWLAN; HEAP, 1978). They present a graphical representation of where potential faults commence and where functional faults begin.

Figure 2 illustrates a generic PF curve, showing that early detection of failures is beneficial for system safety, integrity, and financial cost. It is important to note that the curve pertains to incipient failures rather than abrupt ones (see Subsection 2.1.1). Additionally, the means of detecting faults vary depending on the type of monitored system. Ultrasound is generally used to detect faults in their early stages, allowing for predictive system maintenance. Meanwhile, utilizing methods like vibration, thermography, and oil analysis, particularly for rotating systems, enables the detection of faults at a suitable stage for scheduled preventive or corrective maintenance. If a fault remains undetected at these stages, it may develop and display observable signs, including noise, high temperature, fluid discoloration, and loose or damaged parts. If this occurs, the fault could progress to the point where the system's operation is halted (functional failure), necessitating corrective intervention or even replacement.

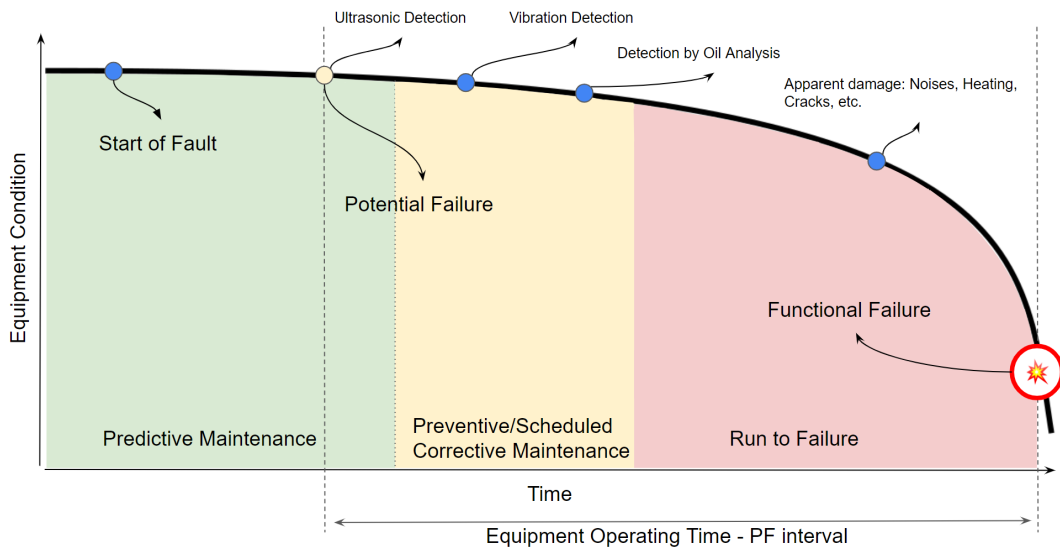


Figure 2 – Generic potential failure curve (PF curve)

From the analysis presented above, it is evident that monitoring physical systems for detecting and diagnosing faults is a vital area of academic and commercial interest, leading to extensive research on the subject with varying degrees of fault analysis and approach, which are described in the subsequent sections.

2.1.1 Nomenclature

Since its inception, a divergence has occurred in the terminology employed in the literature for fault detection and diagnosis. With this in mind, the committee of SAFEPROCESS, the *Symposium on Fault Detection Supervision and Safety in Technical Processes*, organized by the *International Federation of Automatic Control* (IFAC), proposed a catalog of definitions in 1991 about the subject of fault diagnosis and fault-tolerant control. SAFEPROCESS is a highly significant international event for specialists in academia and industry, concentrating on topics such as dynamic system diagnostics, fault-tolerant control, process supervision, system reliability, safety, and other related areas.

The definitions below adhere to the established standard in (SILVIO SIMANI, 2018) and follow the SAFEPROCESS guidelines.

- **States and Signals**

Fault: is an unpermitted deviation of at least one characteristic property or parameter of the system from acceptable, usual or standard conditions.

Failure: a permanent interruption of a system's ability to perform a required function under specified operating conditions.

Malfunction: is an intermittent irregularity in the fulfillment of the desired function of a system.

Error: is a deviation between a measured or calculated value of an output variable and its true or theoretically correct value.

Disturbance: is an unknown and uncontrolled input acting on a system.

Residue: is an indicator of fault, through a deviation between measurements and calculations based on system model equations.

Symptom: is a change in observable behavior in relation to normal behavior.

- **Functions and Tasks**

Fault Detection: is the determination of the presence of faults and the time of occurrence.

Fault Isolation: is the determination of the type, location and time of detection of a fault. Fault detection follows.

Fault Identification: is the determination of the size and time-varying behavior of a fault. It follows fault isolation.

Fault Diagnosis: is the determination of the type, size, location and detection time of a fault. It follows fault detection and includes fault detection and identification.

Monitoring: a continuous real-time task of determining the condition of a physical system, recording information, recognizing and indicating anomalies in behaviour.

Supervision: Monitoring a physical system and taking appropriate action to maintain operation in the event of failure.

- **Approaches**

Data-Driven Approach: establishes a model that provides the mathematical relationships between the system's variables inferred on the basis of data from the system itself.

Model-Based Approach: establishes a model that provides the mathematical relationships between the system's variables derived on the basis of the physical laws that describe the system's own behavior.

- **System Properties**

Reliability: is the ability of a system to perform a required function under given conditions, within a given scope, over a given period of time.

Safety: is the ability of a system not to cause danger to people, equipment or the environment.

Availability: is the probability that a system or piece of equipment will operate satisfactorily and effectively at any point in time.

- **Fault Types**

Abrupt Faults: is modeled as a step function and represents a bias in the monitored signal.

Incipient Faults: is modeled using ramp signals and represents the deviation of the monitored signal.

Intermittent Faults: is a combination of pulses with different amplitudes and lengths.

- **Fault typology**

Additive Fault: influences a variable by an addition to the fault itself. It can represent, for example, deviation in a sensor.

Multiplicative Fault: are represented by the product of a variable with the fault itself. They can appear as parameter changes within a process.

In this thesis, the above definitions are adopted, except that faults and failures are considered undesirable situations regardless of whether the system can perform its function.

2.1.2 Approaches to Fault Detection and Diagnosis

There are several approaches to fault detection and diagnosis, the most important of which are the model-based and data-driven approaches ([MANSOURI; HARKAT, et al., 2020](#)). This section discusses these approaches and their characteristics and tools.

2.1.2.1 Hardware Redundancy

Numerous applications employ a fault detection and diagnosis approach that utilizes multiple sensors, actuators, or components to measure or control a signal. This allows the diagnosis of faults by comparing various redundant hardware information. This approach is known as the physical or hardware redundancy method, where a voting technique is utilized to determine the occurrence and location of the fault. While hardware redundancy can be highly effective, it may pose challenges, such as increased expenditures associated with equipment, maintenance operations, and storage space. These factors can render this solution unfeasible ([SILVIO SIMANI, 2018](#)).

2.1.2.2 Model-Based Approach

One option to reduce costs while preserving system reliability is to generate a model of the system under investigation, allowing for the estimation of a specific variable using existing hardware, thus avoiding the need for additional hardware. This approach enables

the examination of the residual between the model's predictions and the real system's measurements, enabling the detection of faults via logical analysis. This method is identified as analytical redundancy, frequently referred to as a model-based approach (SIMANI; FANTUZZI; PATTON, 2003).

The great advantage of this approach is the ability to implement the model using software, which offers flexibility and cost savings. Another benefit is that an accurate analytical description of the system with the model can improve the efficiency of detecting faults and enable quick recovery from disturbances and unknown variables impacting the system. The effectiveness of this approach depends on the quality of the model used, which represents a challenge due to the limitations that may exist in providing an accurate analytical description of the system. Thus, an unreliable description will lead to a drop in the accuracy of the estimates generated and disturbances and noise affecting the monitored system, which can lead to false alarms or undetected faults.

2.1.2.3 Data-Driven Approach

Sometimes, obtaining precise models of analytically-described monitored systems proves challenging, particularly for complex systems. In instances where only data input and output measurements are available, an alternative method must be employed to facilitate fault detection, such as the data-driven approach (MANSOURI; HARKAT, et al., 2020). This approach relies on analyzing a past data set acquired from the system. Overall, the data is utilized to generate an empirical system model that will then be implemented to perform the fault detection task. The task of obtaining data-based models is called *System Identification*, with emphasis on black-box identification, which is based only on data (Section 2.2).

Like the model-based method, the data-based approach has the essential advantage of being implementable through software. This feature provides ease of use, flexibility, and affordability. Furthermore, this method relies on data gathered from the monitored system, thus avoiding dependency on a comprehensive understanding of the physical principles governing it. This circumstance may be unattainable, as previously mentioned. However, there are instances in which data and knowledge of these physical principles are combined, resulting in using identification techniques known as grey-box. Given the previously mentioned characteristics and the fact that many physical systems are considered black-box systems, a data-based approach becomes highly desirable for monitoring solutions that aim for fault detection and diagnosis. Strategies such as residue analysis and the use of classifiers can be employed for this purpose (MANSOURI; HARKAT, et al., 2020; SILVIO SIMANI, 2018). To achieve this goal, various methods are utilized, including regression via the *Partial Least Squares* (PLS) (JIA; DU; ZHANG, Y., 2016), PCA (BENCHEIKH et al., 2020), *Independent Component Analysis* (ICA) (ŽVOKELJ; ZUPAN; PREBIL, 2016), *Canonical Variable Analysis* (CVA) (VENKATASUBRAMANIAN et al., 2003), *Artificial Neural Networks* (ANN)

(JANSSENS et al., 2016), fuzzy systems (GUO; FAN; AHN, 2020) and SVM (SANTOS et al., 2015).

Despite their potential, data-based approaches encounter challenges. These techniques' success relies on the obtained model's quality, which, like in the model-based approach, depends on the quality of data employed. Insufficient information in the data is a well-known hurdle in extracting the correct dynamics of the system (AGUIRRE, 2015). The quality of sensors can influence measurements and may be affected by disturbances and noise, which makes identifying the desired model a challenging task. Finally, system identification techniques can be computationally expensive, particularly for complex systems, making it difficult to implement them in constrained systems, such as embedded systems. The subsequent subsections will delve deeper into these issues.

2.1.3 Residue Analysis

As subsection 2.1.2.2 mentions, fault detection relies on residue analysis in the model-based approach. However, this feature can also be utilized in the data-based approach, even though the model is obtained from the system's data. In turn, the residual should be minimal when the system is under normal operating conditions, without any faults. In an ideal scenario, the residual signal should be zero; however, this is seldom the case in practice due to uncertainties in the model and noise in measurements. Nonetheless, during the occurrence of a fault, we expect this value to change significantly, indicating a varying system operating condition.

In residual analysis, the success of this strategy is influenced by three fundamental factors: the quality of the model, the quality of the residual generation, and the strategy used to evaluate these residuals (SIMANI; FANTUZZI; PATTON, 2003). The first two factors are directly interconnected as good models result in sound residual generation, while poor models diminish the confidence in the generated residuals. The quality of measurements in the real system is another crucial element in generating residues, as it can be impacted by sensor quality and high noise signals. Notably, the generation of residuals is not limited to the simple disparity between estimated and measured outputs, and metrics like the coefficient of determination R^2 are commonly employed.

When analyzing generated residues, a feasible implementation strategy would be to set a limit value for the alteration in the residue signal. If this value is exceeded, a fault will be detected. This approach, called geometric analysis, is commonly employed due to its simplicity and efficacy in multiple monitoring situations, particularly for identifying sudden faults (MOUSSA et al., 2022). Nevertheless, certain weaknesses exist, including susceptibility to false alarms (false positives) due to noise, input variations, and changes in the monitored system's operating point. As previously stated, the residual analysis approach is effective for sudden faults. However, incipient faults may not substantially alter the behavior of the

actual system, particularly during its initial phases. They may not produce a deviation in the residual signal adequate for detection. Additionally, there may be cases where such changes are concealed by control actions or recursive modifications to the model, such as recursive parameter estimation, that maintain the residuals below the alarm threshold. In these instances, such failures would remain undetected, resulting in a false negative conclusion.

Achieving high accuracy and precision in hardware implementation requires capturing sufficient system dynamics to generate dependable residuals for fault detection, which can be computationally demanding, time-consuming, and energy-intensive. Nevertheless, a significant challenge lies in deploying such models in embedded systems that require online monitoring and re-identification of the monitored system. In such instances, the objective is typically to strike a balance between model precision and accuracy, and the performance of the hardware, considering the cost-effectiveness of implementing the solution, while considering embedded system limitations.

A combination of residual analysis and other fault analysis and detection techniques, including monitoring parameter variations, are adopted to address these issues. This approach aims to deliver a practical and feasible solution to embedded-based monitoring systems. Among these techniques, control charts are used and discussed further in subsection 2.1.4 (HOFFMAN, 2019). Finally, notwithstanding the weaknesses mentioned above, residual analysis is frequently employed for fault detection. This includes certain fault classification methods, such as using residual signals in the design of classifiers by neural networks (JIAYANG et al., 2023).

2.1.4 Control Charts

The preceding section outlined the features of fault detection based on residual analysis. It was demonstrated that this technique has specific weaknesses and a practical option for enhancing its efficacy, particularly in analyzing parameter fluctuations, is using control charts. These charts referred to as “*Control Charts*”, are based on a Statistical Process Control technique that is frequently employed to monitor the variable outputs of systems. Essentially, this method involves three lines on a graph. The central line denotes the average value of the monitored characteristic (variable), representing the “*in control*” state. Furthermore, there are two other lines: the *Upper Control Limit* (UCL) and the *Lower Control Limit* (LCL). These parameters are carefully selected to ensure the monitored characteristic is within the control state when its samples fall within a specified range. When the samples exceed these limits, it indicates that the monitored feature is in an “*out of control*” state (MONTGOMERY, 2009).

The general model for a control chart can be defined using equation 2.1. In this equation, L represents the distance between the control limits and the center line. Additionally, μ_θ and σ_θ represent the mean and standard deviation of the parameter θ , respectively.

$$\begin{aligned}
UCL &= \mu_\theta + L\sigma_\theta \\
CenterLine &= \mu_\theta \\
LCL &= \mu_\theta - L\sigma_\theta
\end{aligned} \tag{2.1}$$

The selection of the distance between the control limits and the central line holds significant importance in the effectiveness of a control chart. Opting for small values of L would enhance the system's sensitivity to variations, potentially resulting in false alarms. Conversely, larger values would decrease sensitivity, leading to potential failures in detecting variations (false negatives). In practice, control charts typically employ a limit of three standard deviations ($L = 3$), a value justified by its satisfactory performance (MONTGOMERY, 2009). Shewhart originally proposed this value for process monitoring. Assuming the process parameter θ follows a normal distribution, $P(\mu + 3\sigma, \mu - 3\sigma) = 0.9973$, implying that nearly all observed data will fall within three standard deviations from the mean. The probability of θ falling outside this range is only 0.0027, indicating a rare occurrence. The presence of such a low probability event suggests that the process parameter is out of control (ZHANG, J.; HOU; ZHANG, Z., 2022).

A significant drawback of a Shewhart control chart is that it solely relies on the information derived from the most recent observation within a sample, disregarding any insights offered by the entire sequence of data points. Consequently, the Shewhart control chart exhibits relatively low sensitivity to minor process changes, such as those on the magnitude of 1.5σ or less (MONTGOMERY, 2009). An efficient alternative to the Shewhart control chart, handy when the objective is to identify subtle variations in the monitored process, is the cumulative sum (CUSUM) control chart.

The CUSUM chart effectively incorporates all the information in the sequence of analyzed samples by plotting the cumulative sum of deviation values from a reference value. In this context, let C^+ (denote the upper cumulative sum value, accumulating derivations from μ_0 that are above target) and C^- (denote the lower cumulative sum value, accumulating derivations from μ_0 that are below target), θ represents the monitored parameter, K being the experimentally obtained reference value and representing the size of the deviation to be detected and μ_0 and σ signify the mean and standard deviation of the analyzed sample set, respectively; consequently, we can express this relationship as follows, for the high and low CUSUM values:

$$C^+(\theta_k) = \max\{0, C(\theta_{k-1}) + \theta_k - (\mu_0 + K\sigma)\} \tag{2.2}$$

$$C^-(\theta_k) = \max\{0, C(\theta_{k-1}) + (\mu_0 - K\sigma) - \theta_k\} \tag{2.3}$$

An alternative to the CUSUM is the *Exponentially Weighted Moving Average* (EWMA) chart. While the CUSUM gives equal weight to all past observations, the EWMA gives more

weight to the most recent observations. The CUSUM and EWMA techniques have similar performance in detecting small data deviations, with the latter being more manageable to set up and operate (MANSOURI; HARKAT, et al., 2020).

Based on the above considerations, this work adopts a control chart model called CCGPU (*Control Chart on GPU*), to examine parametric variations of the NARX model. Specifically, the proposed solution combines a *moving average* (MA) filter with CUSUM. The main objective is to use the MA filter as a low-pass filter to smooth out parameter fluctuations, such as noise, thus enabling a more reliable trend analysis. Subsequently, the difference between the filtered and unfiltered parameter values is calculated ($e_\theta = \theta_{MA} - \theta$), as this difference is expected to be more pronounced in cases involving variations in the dynamics of the monitored system. Finally, the CUSUM (as represented in the 2.2 and 2.3 equations) is applied to the e_θ signal. Details of this implementation will be dealt with in the chapter 5.

2.1.5 SVM Classifiers

Support Vector Machines were developed by Vapnik (VAPNIK, 1995) and since then have been widely used for classification problems in several applications. The main characteristics of the SVM are a) flexibility of kernel choice, b) performance in problems that have little training data, c) it has few hyperparameters, d) the training problem is a quadratic and convex optimization problem and, therefore, it has a global optimal solution. (see Equation 2.4).

In short, SVMs are a type of classifier/regressor that involves a convex optimization problem when calculating their parameters, such as the Lagrange coefficients (VAPNIK, 1995). Obtaining SVM parameters entails solving a quadratic optimization problem, which ensures a global optimal solution for a predetermined set of hyperparameters. This attribute is highly appealing when selecting a classification model.

$$\begin{aligned}
 \text{Max } L_d(\alpha_i) &= \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\
 \text{s.t.} \\
 \sum_{i=1}^N \alpha_i y_i &= 0 \\
 \xi_i &\geq 0 \quad i = 1, \dots, N \\
 0 \leq \alpha_i &\leq C \quad i = 1, \dots, N
 \end{aligned} \tag{2.4}$$

where C is a regularization parameter, N is the training set cardinality, α are the Lagrange multipliers, ξ_i are the slack variables and $K(\cdot)$ is the symmetric *Positive Semi-definite* (PSD) matrix. This matrix must satisfy Mercer's conditions and is called kernel matrix or kernel function (BURGES, 1998).

The optimal solution to the optimization problem of equation 2.4 is a sparse vector of

weights (the Lagrangian coefficients) associated with each training set element (see equation 2.5).

$$\hat{f}(\mathbf{x}) = \text{sgn} \left(\sum_{i \in SV} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + \rho \right) \quad (2.5)$$

where $\hat{f}(\mathbf{x})$ is the predict value of the vector features $\mathbf{x}$, SV is the support vectors set, α are the Lagrangian coefficients, $\mathbf{s}_i$ is the i -th training vector and ρ is the bias. Thus, the greater the sparsity of the optimal vector, the lower the complexity of the SVMs.

This work picked the radial basis function and polynomial kernels as SVM kernels. The RBF kernel is the most used in many miscellaneous application areas and has the ability to weight data locally, thus having a great capacity to adjust to data from different applications. However, as a disadvantage, the parameter γ , which controls the local adjustment, must be carefully chosen with the regularization parameter (C) to avoid the model overfitting (CERVANTES et al., 2020; SMOLA; SCHÖLKOPF, 1998). The equation 2.6 represents the RBF kernel function.

$$K_{RBF}(\mathbf{x}_i, \mathbf{x}_j) = \exp \left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\gamma} \right) \quad (2.6)$$

where the $\mathbf{x}_i$ and $\mathbf{x}_j$ are the features vectors and the RBF kernel parameter γ is a positive real number.

The polynomial is a kernel in which distinct distant data influence each other, that is, it is a global kernel. Computationally, it is simple to calculate as it only involves addition and multiplication operations, making the SVM model attractive to embedded systems (SMOLA; SCHÖLKOPF, 1998; SMITS; JORDAAN, 2002).

In summary, the polynomial and RBF kernels were chosen due to their efficiency in various applications (CERVANTES et al., 2020), because they belong to different groups of kernels (local and global) (SMITS; JORDAAN, 2002) and, particularly about the polynomial kernel, due to their simplicity computational with a view to application in embedded systems. The equation 2.7 represents the polynomial kernel function.

$$K_{poly}(x_i, x_j) = (\beta \mathbf{x}_i \cdot \mathbf{x}_j + \kappa)^d \quad (2.7)$$

where the β and κ are kernel parameters, adopted as constant equal to one in this work, and d integer positive number that must be chosen.

We seek SVM with the least complexity possible and with the greatest accuracy. Both the complexity and accuracy of an SVM are controlled indirectly by the regularization parameter (C) (see Eq. 2.4) and the kernel parameters, which together are called *hyperparameters*. However, finding the hyperparameters that induce SVMs with low complexity

and high accuracy is a complex problem, because each dataset and kernel have a specific optimal configuration. This optimization problem is called *Hyperparameter Optimization* (HPO) (FEURER; HUTTER, 2019).

The HPO was modeled as *Multi-objective Optimization Problem* (MOOP) because we are interested in SVMs models that have high accuracy and low complexity (number of support vectors). In addition, SVMs with few support vectors have a lower probability of overfitting.

The NIOTS was the selected tool to find near-optimal solutions for the HPO. This system has implemented the metaheuristic *Adaptive Parameter control with Mutant Tournament Multi-Objective Differential Evolution* (APMT-MODE). The APMT-MODE has an adaptive mechanism that pursues adjustment to each kernel and dataset HPO combination (SILVA SANTOS et al., 2021). The polynomial and RBF kernels have very different characteristics in the way they evaluate the similarity between the data, as well as their complexity about the implementation in embedded systems. Therefore, in this study, both experiments were carried out.

The SVMs are separating data hyperplanes in a high dimensional space where the data are linearly separable (Hilbert space $\mathcal{H}$) and, therefore, the SVMs given by Equation 2.5 are binary classifiers. The fault classification problem includes thirteen classes, which require an SVM set where each member of the set acts on a binary classification subproblem.

One-Against-One (AOA) is the methodology adopted by the authors of the LibSVM tool to solve multiclass classification problems (HSU, C.-W.; LIN, 2002). AOA splits the multiclass classification problem with k classes into $m = \frac{k(k-1)}{2}$ binary classification problems, and, for each of the subproblems, a binary classifier is generated. The result of the multiclass classification consists of the most voted class by the binary classifier models. Figure 3 represents the scheme for a three-class problem.

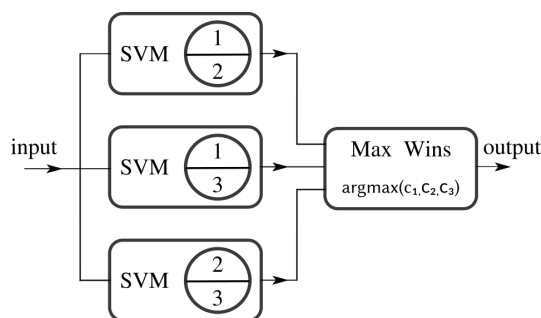


Figure 3 – The three classes classifier One-Against-One example with Max Win aggregation methodology.

This work used the “*Max Wins*” aggregation method among several other existing ones (GALAR et al., 2011). In the “*Max Wins*” aggregation method, each ensemble member votes for a class, and the class that receives the most votes wins. The OAO, compared to the

One-Against-All technique, is less sensitive to unbalanced classes and the training process is faster (HSU, C.-W.; LIN, 2002).

2.2 Systems Identification

As outlined in subsection 2.1.2.3, system identification plays a crucial role in the data-driven approach to creating an empirical model that accurately represents the monitored system. To be more precise, system identification is the process of obtaining a dynamic model of a system through measurements of its inputs and outputs, which serves as a complement to the simulation problem (AGUIRRE, 2015). This technique can be seen as the interface between the real world of applications and the mathematical world through a model that can emulate the system, predict its response to certain inputs, and investigate different design scenarios (LJUNG, 2010). Therefore, identifying system dynamics is an essential task for various areas of research that deal with real systems for monitoring and fault detection and areas such as system control and simulation.

System identification techniques can use prior knowledge about the behavior or physical principles that govern the system, combining this knowledge with the system's measured data. In this case, the identification is said to be of the gray-box type. Another approach is to consider the system as a black-box, with little or no prior knowledge about it, and take only its data into account, thus performing a black-box identification. In addition, about the models obtained from system identification, there is a variation between non-parametric models, which do not process any specific structure, but are generally described by non-parameterized responses, and parametric models, which process a specific structure and are characterized by delay, order and a set of parameters (BILLINGS, 2013).

Many physical systems exhibit nonlinear dynamics that defy the superposition principle, classifying them as nonlinear systems. This class of systems is typically more intricate than linear, necessitating more complex identification techniques for capturing nonlinear dynamics. The early investigations into system identification for nonlinear systems were primarily dominated by techniques that relied on the Volterra series, an elongation of the linear convolution integral that represented moderately nonlinear systems as a sequence of multiple summations (BILLINGS, 2013).

As research within the field progressed, numerous methodologies have emerged to tackle the issue of system identification, with many concentrating on applying neural networks and bioinspired algorithms. NARMAX/NARX models provide an alternative approach to parametric representation to represent a wide range of nonlinear systems (BILLINGS, 2013). However, the effectiveness of these models is closely tied to the choice of model structure, which is a critical aspect of the identification process. This task may be accomplished by implementing the FROLS algorithm, a standard technique for detecting

nonlinear model structures by categorizing significant terms, as discussed in (CHEN, S.; HONG X. AD HARRIS; SHARKEY, 2004) and (AKINOLA et al., 2019). More information regarding this approach can be found in subsection 2.2.1.

Within this context, the nonlinear characteristics of most real systems have led to the development of techniques with high computational complexity, especially for black-box identification, which can make them unfeasible to use in systems with restrictions, such as embedded systems. In this case, there have not yet been many studies on the best way to map nonlinear system identification techniques to embedded system platforms that may have parallelism capabilities, such as those that incorporate GPUs (see section 3.2).

2.2.1 NARMAX/NARX Polynomial Model

This section describes the NARMAX polynomial model, whose number of terms in the complete model is reduced using the FROLS algorithm. The aim of identifying nonlinear systems with the NARMAX model is to obtain a mathematical abstraction of the real system based solely on input-output data, which was introduced in (BILLINGS; LEONTARITIS, 1981) and developed in subsequent publications (BILLINGS; FAKHOURI, 1982; BILLINGS; VOON, 1984). It is worth noting that the NARX model used in this work is a simplification of the NARMAX model and does not contain the moving average part. In this sense, by detailing the NARMAX modeling, the NARX modeling is also detailed. The choice of NARX was based on the need to simplify the solution due to the objective of implementing it in embedded systems, in addition to the good performance of NARX models confirmed in the literature (see section 3.2).

The model can be defined as

$$\begin{aligned} y(k) = F[& y(k-1), y(k-2), \dots, y(k-n_y), \\ & u(k-1), u(k-2), \dots, u(k-n_u), \\ & e(k-1), e(k-2), \dots, e(k-n_e)] + e(k), \end{aligned} \quad (2.8)$$

where $F[\cdot]$ is a nonlinear function and $y(k)$, $u(k)$, and $e(k)$ are respectively system outputs, inputs, and errors at time k . In addition, the model error is described as

$$e(k) = y(k) - \hat{y}(k|k-1). \quad (2.9)$$

Several linear and nonlinear structures can be seen as special cases of the NARMAX model, such as block-structured models, AR/ARX, ARMA/ARMAX (*Autoregressive Moving Average/Autoregressive Moving Average with Exogenous Inputs Model*), and Volterra (BILLINGS, 2013), further increasing the usability of this representation. The most commonly used NARMAX model is the power-form polynomial representation in the form of

an autoregressive model (BILLINGS, 2013)

$$\begin{aligned}
 y(k) = & \theta_0 + \sum_{i_1=1}^n \theta_{i_1} f_{i_1}(x_{i_1}(k)) \\
 & + \sum_{i_1=1}^n \sum_{i_2=i_1}^n \theta_{i_1 i_2} f_{i_1 i_2}(x_{i_1}(k) x_{i_2}(k)) + \dots \\
 & + \sum_{i_1=1}^n \dots \sum_{i_l=i_{l-1}}^n \theta_{i_1 i_2 \dots i_l} f_{i_1 i_2 \dots i_l}(x_{i_1}(k) x_{i_2}(k) \dots x_{i_l}(k)) \\
 & + e(k),
 \end{aligned} \tag{2.10}$$

where

$$\begin{aligned}
 f_{i_1 i_2 \dots i_m}(x_{i_1}(k) x_{i_2}(k) \dots x_{i_m}(k)) = & \theta_{i_1 i_2 \dots i_m} \prod_{k=1}^m x_{i_k}(k) \\
 & , 1 \leq m \leq l
 \end{aligned} \tag{2.11}$$

where l is the degree of the nonlinear polynomial, θ are the model parameters, and $n = ny + nu + ne$ is the sum of the delays used for the outputs, inputs, and noise respectively. Note that to have a more general representation, in equation 2.11, $x_{i_k}(k)$ can represent inputs (u), outputs (y) or errors (e) at $t = k$ (WEI, H.-L.; BILLINGS, 2003). That is, $x_m(k) = y(k - m)$, if $1 \leq m \leq ny$, $x_m(k) = u(k - (m - ny))$, if $ny + 1 \leq m \leq ny + nu$, and $x_m(k) = e(k - (m - ny - nu))$, if $ny + nu + 1 \leq m \leq ny + nu + ne$, where $x_m(k)$ represents each of the variables in the product $\prod_{k=1}^m$. The NARX model is the subset of the NARMAX model where there are no moving average terms in Eq. (2.8).

For large values of ny , nu , ne and l , the number of initial candidates M for model NARMAX or NARX can become large and can be calculated by $M = \frac{(n+l)!}{n!l!}$; that is, the number of terms grows combinatorially for achieving the *full-model*. For most practical cases, only a few terms among all candidates are relevant to describing the real system's dynamics, leading to a *parsimonious model* that can generalize the measured input-output data. It is easy to see that as the model increases beyond the true system structure, the model's predictive performance deteriorates by over-fitting phenomena, which usually produce ghost bifurcations in nonlinear systems. In general, equation 2.10 can be represented as

$$y(k) = \sum_{i=1}^M [\theta_i p_i(k)] + e(k) \tag{2.12}$$

where $\theta_i(k)$ ($i = 1, 2, \dots, M$) are the system's parameters to be estimated and $p_i(k)$ is a regressor term formed by some combination of variables $x_{i_j}(k)$ defined in each term of equation 2.10; that is, $p_i(k)$ are the model terms defined as:

$$\begin{aligned}
 p_1(k) = & y(k - m_{y,1}) \dots y(k - m_{y,m_y}) u(k - m_{u,1}) \dots u(k - m_{u,m_u}) \\
 & e(k - m_{e,1}) \dots e(k - m_{e,m_e})
 \end{aligned} \tag{2.13}$$

Note that equation 2.12 can be rewritten in its matrix form as:

$$Y = P\Theta + e(k) \quad (2.14)$$

It can be observed that choosing the terms that will make up the final model is important since one may not obtain enough terms, which will result in under-fitting.

2.2.2 FROLS strategy based on orthogonal decomposition

The FROLS algorithm selects and ranks the most significant terms according to a criterion (e.g., the error reduction ratio). Its working principle is the basic *Orthogonal Least Squares* (OLS) algorithm that selects the most relevant term of the model at each step. A full search of all remaining or unselected model terms is conducted at each step. The algorithm determines which nonlinear terms should be included in the parsimonious model by computing each potential model term's contribution to the system output (BILLINGS S.A., K. M.; S., 1988),(BILLINGS S.A., C. S.; M.J., 1989). OLS starts from the initial model defined in equation 2.12 and introduces an auxiliary and equivalent model:

$$y(k) = \sum_{i=1}^M g_i q_i(k) + e(k) \quad (2.15)$$

where column vectors $q_i(k)$ ($i = 1, 2, \dots, M$) are orthogonal over the estimation data set of N samples in the sense that

$$\sum_{k=1}^N q_i(k) q_j(k) = \begin{cases} d_i, & i = j \\ 0, & i \neq j \end{cases} \quad (2.16)$$

with $d_i = \sum_{k=1}^N q_i^2(k) \neq 0$. In this case, let's define a matrix form of the equation 2.15 as:

$$Y = QG + e(k) \quad (2.17)$$

where $G = [g_1, g_2, \dots, g_M]^T$ is an auxiliary parameter vector that can be calculated by the Moore–Penrose inverse as

$$G = (Q^T Q)^{-1} Q^T Y. \quad (2.18)$$

Given that $Q^T Q$ is a diagonal matrix, its inverse is trivial. This fact allows to calculate each element of the vector G by

$$g_i = \frac{\langle \mathbf{y}, \mathbf{q}_i \rangle}{\langle \mathbf{q}_i, \mathbf{q}_i \rangle}, i = 1, 2, \dots, M. \quad (2.19)$$

Additionally, let's assume that in equation 2.14 there is an orthogonal decomposition such that the matrix P can be decomposed as

$$P = QA, \quad (2.20)$$

where $Q_{N \times M}$ has orthogonal columns, according to conditions in equation 2.16 and $A_{M \times M}$ is unit upper triangular matrix. The classic Gram-Schmidt algorithm computes A , column by column, and orthogonalizes P as follows:

$$\begin{aligned} q_1(k) &= p_1(k) \\ q_2(k) &= p_2(k) - a_{1,2}q_1(k) \\ q_3(k) &= p_3(k) - a_{1,3}q_1(k) - a_{2,3}q_2(k) \\ &\vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ q_m(k) &= p_m(k) - \sum_{r=1}^{m-1} a_{r,m}q_r, \quad m = 2, 3, \dots, M \end{aligned} \quad (2.21)$$

where

$$a_{r,m} = \frac{\sum_{k=1}^N p_m(k)q_r(k)}{\sum_{k=1}^N q_r^2(k)}, \quad 1 \leq r \leq m-1 \quad (2.22)$$

From equations 2.14, 2.17, and 2.20 we can infer:

$$Y = (PA^{-1})(A\Theta) + e = QG + e, \quad (2.23)$$

and in this case, it is clear that

$$Q = PA^{-1} \text{ and } G = A\Theta \quad (2.24)$$

Therefore, FROLS firstly calculates the vector G using equation 2.19 and, taking into account that A is an upper triangular matrix; then, it uses the second part of equation 2.24 to obtain the values of Θ recursively (see equations 2.21 and 2.22).

2.2.3 FROLS strategy to get the best terms

The theoretical strategy of FROLS starts from the equation 2.15 and 2.25 obtaining the equation 2.26 in the form of a dot product.

$$y^2(k) = \left(\sum_{i=1}^M g_i q_i(k) + e(k) \right)^2 \quad (2.25)$$

$$\langle Y, Y \rangle = \sum_{i=1}^M g_i^2 \langle \mathbf{q}_i, \mathbf{q}_i \rangle + \langle \mathbf{e}, \mathbf{e} \rangle \quad (2.26)$$

Dividing the equation 2.26 by the dot product $\langle Y, Y \rangle$, we obtain

$$1 = \sum_{i=1}^M \frac{g_i^2 \langle \mathbf{q}_i, \mathbf{q}_i \rangle}{\langle Y, Y \rangle} + \frac{\langle \mathbf{e}, \mathbf{e} \rangle}{\langle Y, Y \rangle} \quad (2.27)$$

Therefore, the Error Reduction Ratio (ERR) is defined by the inclusion of a k -th term by

$$[\text{ERR}]_k = \frac{g_i^2 \langle \mathbf{q}_i, \mathbf{q}_i \rangle}{\langle \mathbf{y}, \mathbf{y} \rangle} \times 100\%, i = 1, 2, \dots, M. \quad (2.28)$$

Additionally, equation 2.28 can be rewritten using equation 2.19 as

$$[\text{ERR}]_k = \frac{\langle \mathbf{y}, \mathbf{q}_i \rangle}{\langle \mathbf{y}, \mathbf{y} \rangle \langle \mathbf{q}_i, \mathbf{q}_i \rangle} \times 100\%, i = 1, 2, \dots, M. \quad (2.29)$$

Finally, the Error-to-Signal Ratio (ESR) is defined as

$$ESR = 1 - \sum_{i=1}^M \text{ERR}_i = \frac{\langle \mathbf{e}, \mathbf{e} \rangle}{\langle Y, Y \rangle}. \quad (2.30)$$

Repeated application of this strategy provides unbiased estimates of each coefficient, indicating each model term's contribution to the system output variance and selecting the essential terms of the full model. The terms selected from the original model will compose a reduced model (the parsimonious model), which will perform well in terms of accuracy, ensuring a reduced model complexity.

2.2.4 Adjusting model parameters via Slide Window Recursive Least Squares

Modifications experienced by the system, whether caused by environmental fluctuations, faults, or other factors, result in a discrepancy between the model's output and the observed system. In such scenarios, it becomes essential to adjust the model to ensure its accuracy in representing the real system. This adjustment can be accomplished through the recursive estimation of parameters. In this regard, the algorithm proposed in this study incorporates a data window within which the algorithms are re-estimated using RLS technique. This specific approach is referred to as SWRLS, being a characteristic tool in the identification of time-varying systems (BILLINGS, 2013).

The RLS algorithm possesses an infinite memory characteristic, meaning that both initial data and recent data contribute to the recursive parameter calculation. However, this feature proves to be less advantageous for time-varying processes as older data may no longer accurately represent the current dynamics of the system. In such cases, it becomes desirable for the most recent samples to carry greater weight in the parameter adjustment. To address this issue, an version of the RLS algorithm, incorporating a forgetting factor, is employed as depicted in equations 2.31 to 2.34. It is important to note that in the equations mentioned above, y denotes the measured output value of the system, ϕ represents the regressor vector, e signifies the error, K represents the estimation gain, $\hat{\theta}$ denotes the parameter vector, P symbolizes the covariance matrix, λ represents the forgetting factor, and k denotes the current time instant.

$$e(k+1) = y(k+1) - \phi^T(k+1)\hat{\theta}(k), \quad (2.31)$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) + K(k+1)e(k+1), \quad (2.32)$$

$$K(k+1) = \frac{P(k)\phi(k+1)}{\lambda + \phi^T(k+1)P(k)\phi(k+1)}, \quad (2.33)$$

$$P(k+1) = \frac{1}{\lambda} [P(k) - K(k)[P(k)\phi(k+1)]^T]. \quad (2.34)$$

Despite incorporating the forgetting factor, previous samples still retain some influence on the parametric reestimation process, albeit with reduced weights. Furthermore, when considering embedded systems, it becomes imperative to account for limitations such as memory constraints. This necessity underscores the importance of adapting the RLS algorithm using a sliding window approach (SWRLS). In this context, a predetermined data window is established within which the RLS algorithm with the forgetting factor is computed. As new data is introduced, the oldest data is discarded, according to (ZHANG, Q., 2000). It is essential to note that the effectiveness of SWRLS relies on appropriately tuning both the size of the sliding window and the forgetting factor, which must be determined based on factors such as the system's characteristics being monitored.

2.3 GPU-Based Embedded Systems

Technological advancements have enabled access to computing resources at any time and place, not only through general-purpose computers or smartphones but also via devices integrated into various products and processes in a manner that remains hidden from the end-user. These devices are commonly referred to as embedded systems. Embedded systems can be defined as hardware components that encompass microprocessors or microcontrollers, memory, and peripheral input/output devices, as well as embedded software. These systems are designed to execute a specific function and are generally integrated into a specific element comprising electronic and/or mechanical parts (SHAOUT; EL-MOUSA; KHALID, 2009).

Developing embedded systems is a vast and multi-applicable area, varying in the platforms and methods used. In general, designers use three well-known approaches to accomplish an embedded task: (i) software approach, (ii) hardware approach, and (iii) hardware/software co-design approach (SHAOUT; EL-MOUSA; KHALID, 2009). The main advantages of software implementations are (a) high flexibility, (b) short development time, and (c) low non-recurring engineering costs. However, on the other hand, fully software implementations are generally slower than hardware implementations, mainly due to von Neumann's bottleneck syndrome (HARTENSTEIN, 2008).

Many platforms have been used to develop embedded systems, ranging from *Application-Specific Integrated Circuits* (ASICs), using microcontrollers, *Digital Signal Processor* (DSP), and *Field-Programmable Gate Array* (FPGA), to platforms with greater application flexibility and widely used in the world of prototyping, such as the widespread Raspberry Pi (GAY, 2014), modules from the ESP family (MAIER; SHARP; VAGAPOV, 2017) and Arduinos (ARDUINO, 2023).

However, there is a class of platforms called MPSoCs that are widely used for more robust applications, mainly involving the need for computational performance and dedicated to parallel applications that encompass artificial intelligence, image processing, and related areas, containing multiple processing elements, usually heterogeneous (WOLF; JERRAYA; MARTIN, 2008). This type of solution is directly linked to the IoT technology and the Edge Computing paradigm.

The term "*Internet of Things*" (IoT) refers to the widespread presence of a variety of devices, such as sensors, actuators, and mobile phones, which, through unique addressing schemes, can interact and cooperate to achieve common goals. The spread of the IoT concept has enabled communication between countless devices, making it an evolutionary milestone in the field and including most future information technology applications (MARWEDEL, 2010). As an example, the exploitation of IoT technology for production has been called "*Industry 4.0*", aiming at more flexible production, for which the entire life cycle, from the design phase onwards, is supported by IoT (BRETTEL et al., 2014).

In turn, the paradigm of edge computing endeavors to relocate programmable devices nearer to where the data is produced or used (SHI et al., 2016). The objective is to connect cloud computing and computing that occurs solely at the data source, providing the following benefits: (a) The aim is to reduce the latency associated with processing, storing, or transmitting data. Additionally, cost savings can be obtained by filtering data and aggregating data sources, thus reducing the cost of cloud and telecommunications. Moreover, simple sensors and non-IP-capable networks can be connected to IP-capable networks and the Internet, enabling efficient communication and networking. (d) The system should ensure computational resilience for critical communications in remote areas and for moving devices, such as mobile robotics. It should also ensure data security and assurance, which may involve anonymizing, deleting, or encrypting data before sending it to the cloud (D. JUDE; MOHAMED; SWATI VIJAY, 2023; SHI et al., 2016; DAI et al., 2019).

These requirements cannot be fulfilled using the current cloud computing infrastructure and, instead, necessitate specific devices. Consequently, edge devices - primarily embedded - can operate as an extension of the cloud or in a more self-governed manner. This area has garnered increased attention in recent years, as Figure 4 depicts.

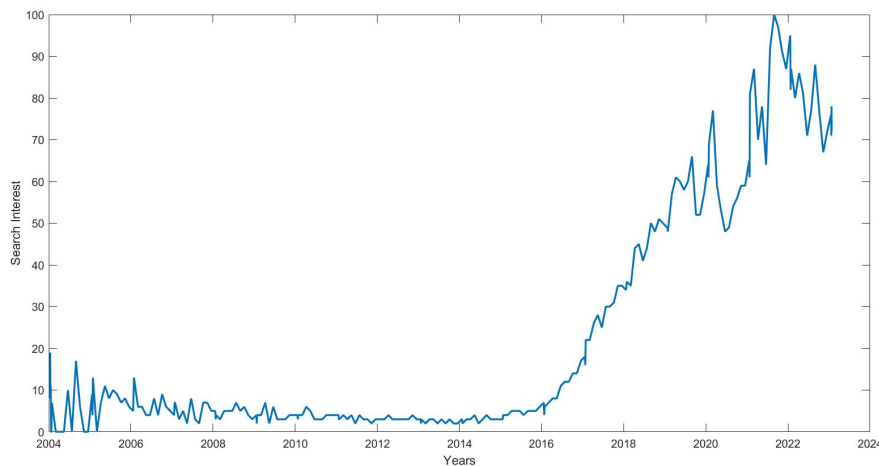


Figure 4 – Interest in Edge Computing over time (data from Google Trends (GOOGLE, 2023)). The number of "search interests" is relative to the highest point on the graph for a given region in a given period. A value of 100 represents the peak popularity of a term. A value of 50 means that the term was half as popular. A score of 0 means that there was not enough data on the term.

Within this context, among the MPSoCs focused on IoT applications and inserted in the edge computing paradigm, we can mention *Parallella* and its architecture *Epiphany* (EPIPHANY, 2016), consisting of the integration of a central CPU as a *Network Processors on Chip* (NOC). Another example is the Xilinx development kits, such as the Versal (VERSAL, 2023) family, combining FPGA structures with an ARM family processor. Some solutions make use of GPUs, such as Xilinx's own Zynq UltraScale devices, which combine GPUs with

dual-core and quad-core ARM processors, as well as NVIDIA's Jetson family of modules (NVIDIA, 2023b), used in this work and presented in greater detail in subsection 2.3.2. It is worth noting that, for this last category of MPSoCs, the massively parallel computing capacity of GPUs and the general-purpose computing capacity of CPUs can enable designers to develop high-performance embedded solutions (HOSSEINABADY; ZAINOL; NUNEZ-YANEZ, 2019).

2.3.1 Graphics Processing Units - GPU

The limitations arising from the high energy consumption and difficulty in heat dissipation by processors due to the clock acceleration of CPUs led to a change in their design approach, increasing the number of cores to improve computing performance through parallel processing (KIRK; HWU, 2012). Thus, the multicore architectures emerged, comprising several processors (cores) encapsulating within the CPU.

Although it is common to refer to any architecture made of several processors as multicore, there exists a differentiation between multicore and manycore devices which varies in approach. Multicore processors use sophisticated cores with *Instruction-Level Parallelism* (ILP) to significantly decrease the execution time of sequential programs. Instances of this architecture include the Intel Xeon W-3175X, which features up to 28 cores, and the Quad-Core ARM Cortex-A57. In contrast, manycore architectures strive to enhance the output of parallel applications by employing numerous cores to carry out multiple operations simultaneously (KIRK; HWU, 2012). Consequently, these architectures integrate less sophisticated cores with a restricted ILP (RAZZOLINI, 2019).

GPUs belong to the manycore architecture category and can operate with thousands of cores simultaneously, delivering superior output than multicore architectures when handling numerous parallel calculations (KIRK; HWU, 2012). Nonetheless, manycore devices require a distinct approach to programming as programs must be organized to utilize the relevant parallelism efficiently. This leads to the technique called parallel programming.

In comparing CPUs and GPUs, there are distinct variations in computing performance that stem from each device's design philosophy. CPUs have a few cores and ample cache memory, allowing them to handle only a few software threads at any given time, with a focus on serial processing. On the other hand, GPUs comprise hundreds of cores that can handle thousands of threads simultaneously, focusing on parallel processing, as shown in Figure 5.

The modern GPU architecture enhances data parallelization, i.e., executing a single instruction on various data items. This architecture is categorized as *Single Instruction Multiple Data* (SIMD) in Flynn's taxonomy (FLYNN, 1972). It comprises of *Streaming Multiprocessors* (SMs), which house several Streaming Processors (*Compute Unified Device Architecture* -

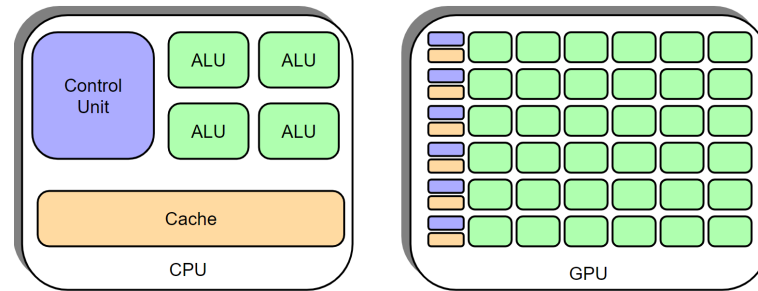


Figure 5 – Comparison between CPU and GPU architecture (Figure adapted from (NVIDIA, 2023a))

CUDA cores) that share control logic and an instruction cache. Furthermore, GPUs possess a main memory that functions as global memory (KIRK; HWU, 2012), together with other levels of memory hierarchy.

From a programming perspective, initially, computations had to be written as a rendering problem to leverage GPU capacity. This programming model imposed several restrictions that complicated GPU usage. To enhance the accessibility of GPUs for other types of problems, GPGPUs were proposed. GPGPUs can be employed to program generic applications and are not restricted to rendering applications. To achieve this objective, the CUDA architecture was proposed by NVIDIA in 2007 (SANDERS; KANDROT, 2010).

The CUDA architecture necessitates the presence of a host (CPU) and one or more devices (GPUs). In this regard, a CUDA program comprises code for the host, composed customarily, and device code employing particular terms to specify it, constituting functions invoked from the host code and executed on the GPU, named kernels. When scripting a CUDA kernel, the number of blocks and threads employed to implement the kernel are indicated. The blocks and threads are arranged into a grid and then allocated to the SMs to execute as many blocks as possible simultaneously, as shown in Figure 6. A limit is in place for the number of blocks its hardware can accommodate on an SM. The scheduler maintains a list of blocks that are still to be executed. After executing a block, an SM is assigned to one of the blocks on the list. (KIRK; HWU, 2012) Thus, the grid comprises blocks, which in turn consist of threads, as illustrated in Figure 7. Additionally, each block and its corresponding threads have distinct identifiers.

The C programming language is the original means by which the CUDA architecture is supported; however, numerous alternatives facilitate its usage in other languages. Numba (NUMBA, 2023), for instance, is a library that directly compiles kernels authored in Python to run on CUDA and supports CUDA GPU programming. Python has become an important language for data science, scientific, and parallel programming. This is due to factors such as its ease of learning compared to languages such as C, and the abundance of libraries that reduce development time. However, its computational performance is inferior to that of C, for example, because Python is an interpreted language. Nonetheless, there has been an

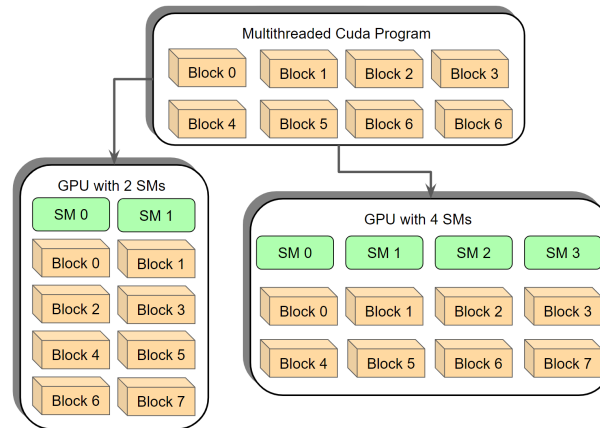


Figure 6 – Automatic scalability of blocks in SMs (Figure adapted from (NVIDIA, 2023a)).

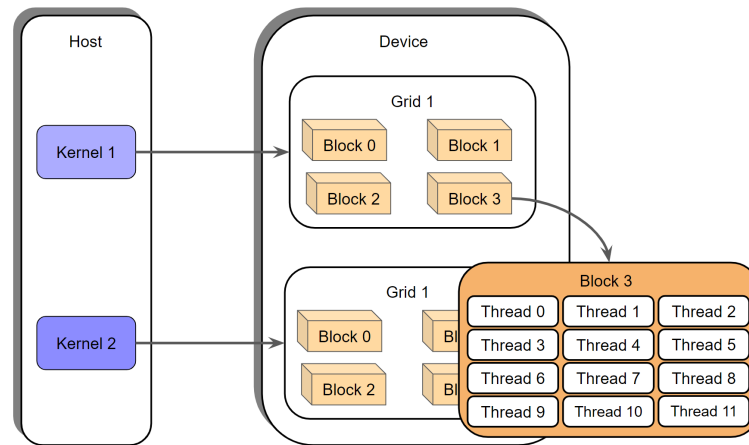


Figure 7 – Hierarchical organization of grids, blocks, and threads (Figure adapted from (KIRK; HWU, 2012)).

increasingly concerted drive to develop software packages such as Dask, PyCompss, and MPI for Python. These packages enable the execution of Python applications on expansive, parallel machines, offering enhanced performance.

Within this context, Numba aims to enable the integration of Python's user-friendliness and flexibility with the enhanced processing speed provided by parallel architectures like the GPU, which are coded in CUDA. The Numba library offers a Just-in-Time (JIT) compiler for CPython - the most widely used Python interpreter written in C. Unlike PyCuda and PyOpenCL, it does not merge C/CUDA code with Python. Instead, it allows developers to create efficient Python-styled applications for CPUs and GPUs. Upon executing a Python script using Numba, the marked functions compile instantly using the LLVM framework. Although Python for GPU programming has a lower computational performance than the C-CUDA language, it can significantly simplify the development of parallel applications. It can achieve up to 85% of the performance attained by using the C language (ODEN, 2020).

Within the preceding discussion, this study utilized the Python language in combi-

nation with the Numba library to create CPU/GPU applications.

2.3.2 Jetson Nano

Nvidia Corporation is a company specializing in the development of electronic components, and, although it is known to the general public for its investment in the computer hardware market, with an emphasis on video processing cards with GPUs, it also operates in the area of supercomputers and solutions for embedded devices and edge computing. Within this field, as a prominent player in the artificial intelligence market, the company has produced a range of modules known as Jetson ([NVIDIA, 2023b](#)). The product line currently comprises 7 different series of devices, each with 17 distinct variations.

In summary, the Jetson modules have been developed to deliver high-performance computing to AI applications at the edge and IoT, while boasting a compact design and low power consumption. Additionally, they are equipped with the *NVIDIA JetPack SDK* development package, which aims to enhance and accelerate application development. The various versions are presented as MPSoCs, with an ARM-based processor and GPUs. They differ in their component capacities and technical specifications like size, cost, computing capability, and memory. Among the models, Jetson Nano is the most basic one ([NVIDIA, 2019](#)).

Currently costing around US\$99 dollars, the Jetson Nano is a small computer comprising a quad-core 1.43 GHz ARM A57 CPU, a 128-core Maxwell GPU, consumption of 5W to 10W and versions ranging from 4 to 2 GB of 64-bit 25.6 GB/s LPDDR4 memory. In this sense, it is a highly cost-effective solution for edge applications involving parallel programming and AI and IoT applications, such as ([SIDDIQUE et al., 2023](#)), ([MITTAL, 2019](#)).

For this work, the Jetson Nano platform was selected due to its advantageous cost-benefit ratio and the capabilities of GPU-based embedded systems for proficient and timely algorithm development, particularly those based on complex linear algebra, like system identification and deep learning models. These factors support its employment to identify embedded solutions for the NARMAX/NARX system via the parallelization of the FROLS algorithm.

2.4 Final Chapter Remarks

This chapter presents the theoretical background of tools used to develop this thesis. As discussed in section 1.6, this work proposes modules based on GPUs to enable a fault detection and diagnosis solution applicable to embedded systems. In this regard, section 2.1 presents, among other concepts, the data-driven approach used (subsection 2.1.2.3), using input and output data from the monitored system to identify its dynamics (subsection 2.2)

using a NARX polynomial model (subsection 2.2.1) whose parsimonious polynomial terms are defined by the FROLS algorithm (subsection 2.2.2) according to the error reduction rate criterion (subsection 2.2.3). The solution includes a mapping of FROLS onto GPUs (subsection 2.3.1), favoring its application in embedded systems (section 2.3).

Once we have a parametric model that represents the monitored system, we can use the analysis of the parameters variation for fault detection, together with classic residual analysis (subsection 2.1.3), using the control chart tool (subsection 2.1.4). This approach, together with a recursive estimation of these parameters via SWRLS (subsection 2.2.4), allows the creation of a monitoring algorithm for fault detection. It is worth noting that implementing the parametric variation detection module, using a control chart in GPUs, improves the solution's computational performance and energy expenditure, favoring its application in embedded systems (section 2.3).

On the other hand, to isolate faults (subsection 2.1.1), an SVM can be used (subsection 2.1.5) whose features are the parameters of the NARX model itself, making it possible to classify faults that have occurred. Similarly, by combining the GPU-based SVM module with the system re-identification module and recursive parameter estimation via SWRLS, a monitoring solution for fault detection and diagnosis applied to embedded systems is obtained.

3 Related Works

CONFIDENTIAL

4 GPU-based Approach for Embedded NARX/FROLS System Identification

CONFIDENTIAL

5 GPU-Based Embedded Monitoring for Fault Detection in Time-Varying Nonlinear Systems

CONFIDENTIAL

6 Embedded SVM-based Classifier for Fault Detection and Diagnosis

CONFIDENTIAL

7 Conclusions

In general, the results reported in this thesis show the feasibility of an embedded solution based on GPUs for monitoring time-varying nonlinear systems to detect and diagnose faults, highlighting the benefits obtained through GPU mapping in terms of computational cost reduction and energy consumption. In this sense, modules for system identification (FROLS_GPU), fault detection (CCGPU), and fault classification (SVM_GPU) were proposed, which proved to be effective in terms of performance, presenting a better computational cost and energy consumption when compared to their standard serial solutions.

In this regard, bearing in mind that system identification is a fundamental part of monitoring, the FROLS_GPU algorithm developed in Chapter 4 offers a solution for this task that is feasible to be embedded, helping to fill a gap in the literature in studies on this topic (see Section 3.2). The results showed the efficiency obtained with the mapping strategy on GPUs and the model's good adherence to the identified system, even when applied to identifying a real system. At the same time, there is a considerable gain in computational terms and energy consumption compared to the serial version. These characteristics are fundamental when considering solutions that require online re-identification of systems in the context of edge computing, which is common in monitoring applications. All these factors help confirm the first hypothesis described in Section 1.3.

In the same sense, the second hypothesis is confirmed in Chapter 5, in which the SWRLS_FROLS_GPU algorithm is proposed as an embedded system monitoring solution for fault detection, helping to fill the gap in the literature on studies in this area (see Section 3.3). The GPU parametric variation analysis mapping via *control chart* (CCGPU) proved efficient, presenting significant improvements in computational time and energy consumption compared to the serial version of the same solution. It is worth mentioning that, based on the research, no studies were found exploring GPU-based control chart solutions.

The results presented in Chapter 6 confirmed the third hypothesis described in Section 1.3, showing the effectiveness of using parameters from the NARX model as features of an SVM (SVM_GPU module) for fault detection and diagnosis. It was confirmed that using all the NARX model parameters is unnecessary to achieve good accuracy. In this sense, although the algorithm provided a correct classification for different feature configurations (by altering the number of features used and the presence or absence of temperature), the complexity of the SVM and, consequently, the execution time and energy consumption are altered. Therefore, the analysis allows a choice of the configuration that best suits the application's requirements under study, considering the number of parameters used, the use of temperature information as features, and the type of SVM kernel. In addition, the

kernel mapping in GPUs reduced processing time and energy consumption compared to their serial versions, which favors the solution implementation in embedded systems.

In summary, the fault detection (SWRLS_FROLS_GPU) and fault detection and diagnosis (SWRLS_SVM_FROS) algorithms, composed based on the modules mentioned above, proved viable when validated with real wind turbine data. This relevant and current application combines the characteristics of a nonlinear time-varying system due to faults and environmental factors, which greatly needs embedded edge computing solutions, especially for offshore installations. In this sense, the chosen case study explores the proposed solution, showing the feasibility of embedded GPU-based monitoring for fault detection and diagnosis in time-varying nonlinear systems. In this way, the proposed solutions confirmed the hypotheses presented in Section 1.3, presenting the advantages of having high flexibility, short development time, and low non-recurring engineering costs while demonstrating positive results regarding system identification and fault detection and diagnosis.

Despite using a specific case study, the solutions mentioned above are flexible, not limited only to this problem, and can be applied to a wide range of linear and nonlinear systems, time-varying or not. In this sense, it is only necessary to tune parameters such as the number of delays in the outputs, inputs, the degree of the model used, and the size of the sliding windows. Therefore, analyses regarding different configurations of these solutions, presented in detail, provide an initial and general notion of their capacity, mainly in computational and energy performance. These discussions above are fundamental for the context of embedded systems, as they provide a clearer view of the solution's capacity conditional on configuration choices.

7.1 Future Works

The development of embedded solutions for system monitoring is an area that is still little explored, with little works involved in this context, despite the great growth potential, given advances in technologies such as IoT and edge computing. In this sense, the work presented in this thesis opens the way for a wide variety of subsequent works, and can thus serve as a basis for new developments in the area.

During the development of this thesis, it was verified that the configuration of the proposed modules is fundamental to their performance. In this sense, several parameters must be tuned, such as the number of system input and output delays, the degree of the polynomial, the number of terms in the final model (given by a limiting value), and the size of the sliding windows used. Tuning varies with the application, but there is a hypothesis that online tuning that adapts to variations in system dynamics could improve results. Therefore, work can be developed to propose online optimization algorithms to tune the parameters of the proposed solution.

When monitoring wind turbines, the most common monitoring solutions are limited to acting simply, stopping the system if faults are detected, and waiting for operation and maintenance services. In this sense, solutions capable of acting on the system could be developed, such as fault-tolerant control applications, which would act in conjunction with the fault detection and diagnosis solution proposed here. Furthermore, some studies show an improvement in the detection and diagnosis of faults in wind farms when comparing data from one turbine with neighboring turbines. Thus, the solution proposed here could be expanded to include this comparison through a network of monitoring devices embedded in IoT.

Finally, all modules based on GPUs developed in this thesis were implemented in Python and the Numba library, which, despite the advantages of short development time and low non-recurring engineering costs, present a performance in computational terms that is inferior to that achieved with C- CUDA, for example. In this sense, translating the codes designed here into the C-CUDA language would be an interesting front of work, expecting even greater performance in terms of computational cost and energy consumption and having comparative bases between the two solutions.

In the same sense, another possibility of work would be to change the approach, developing solutions equivalent to those presented here, but in hardware, such as FPGAs. This would enable a general comparative analysis, not only in terms of language, but also in terms of approach to the problem.

References

- A. JANOT, M. G.; BRUNOT, M. Data Set and Reference Models of EMPS. **I2019 Workshop on Nonlinear System Identification Benchmarks, Eindhoven, The Netherlands**, Apr. 2019. Cit. on pp. [93](#), [177](#).
- AFSHARI, A.; MULUMBA, T.; YAN, K.; SHEN, W.; NORFORD, L. Robust model-based fault diagnosis for air handling units. **Energy and Buildings**, v. 86, p. 698–707, Jan. 2015. Cit. on pp. [30](#), [81](#), [82](#), [85](#).
- AGASTHIAN, A.; PAMULA, R.; KUMARASWAMIDHAS; A., L. Fault Classification and Detection in Wind Turbine Using Cuckoo-Optimized Support Vector Machine. **Neural Comput. Appl.**, Springer-Verlag, Berlin, Heidelberg, v. 31, n. 5, p. 1503–1511, May 2019. ISSN 0941-0643. Cit. on pp. [83](#), [85](#), [146](#).
- AGRAHARI, S.; SINGH, A. K. Concept Drift Detection in Data Stream Mining : A literature review. **Journal of King Saud University - Computer and Information Sciences**, v. 34, 10, Part B, p. 9523–9540, 2022. ISSN 1319-1578. Cit. on p. [23](#).
- AGUILAR-LEAL, O.; FUENTES-AGUILAR, R.; CHAIREZ, I.; GARCÍA-GONZÁLEZ, A.; HUEGEL, J. Distributed parameter system identification using finite element differential neural networks. **Applied Soft Computing**, v. 43, p. 633–642, 2016. ISSN 1568-4946. Cit. on p. [68](#).
- AGUIRRE, L. **Introdução à Identificação de Sistemas**. Jan. 2015. P. 730. ISBN 978-85-423-0079-6. Cit. on pp. [22](#), [23](#), [41](#), [47](#).
- AKINOLA, T. E.; OKO, E.; GU, Y.; WEI, H.; WANG, M. Non-linear system identification of solvent-based post-combustion CO2 capture process. **Fuel**, v. 239, p. 1213–1223, 2019. Cit. on pp. [48](#), [67](#).
- ALTINORS, A.; YOL, F.; YAMAN, O. A sound based method for fault detection with statistical feature extraction in UAV motors. **Applied Acoustics**, v. 183, p. 108325, 2021. ISSN 0003-682X. Cit. on pp. [82](#), [85](#).
- AMIN, M. T.; KHAN, F.; IMTIAZ, S. Fault detection and pathway analysis using a dynamic Bayesian network. **Chemical Engineering Science**, v. 195, p. 777–790, 2019. ISSN 0009-2509. Cit. on pp. [76](#), [77](#).
- AN, Z.; CHENG, L.; GUO, Y.; REN, M.; YANG, Z.; FENG, W.; LING, J.; CHEN, H.; CHEN, W. Edge intelligence in motion: Mobility-aware dynamic DNN inference service migration with downtime in mobile edge computing. **Journal of Systems Architecture**, v. 130, p. 102664, 2022a. ISSN 1383-7621. Cit. on p. [62](#).

- AN, Z.; CHENG, L.; GUO, Y.; REN, M.; YANG, Z.; FENG, W.; LING, J.; CHEN, H.; CHEN, W. Fault diagnosis for nuclear power equipment based on a recursive principal component analysis approach. In: ZHOU, Y.; WU, L.; YANG, Z.; CHEN, Z. (Eds.). **4th International Conference on Informatics Engineering & Information Science (ICIEIS2021)**. SPIE, 2022b. v. 12161. International Society for Optics and Photonics, 121611r. Cit. on pp. 74, 77.
- ANDRÉS; SUÁREZ-CETRULO; QUINTANA, D.; CERVANTES, A. A survey on machine learning for recurring concept drifting data streams. **Expert Systems with Applications**, v. 213, p. 118934, 2023. ISSN 0957-4174. Cit. on p. 23.
- ARCOS JIMÉNEZ, A.; ZHANG, L.; GÓMEZ MUÑOZ, C. Q.; GARCÍA MÁRQUEZ, F. P. Maintenance management based on Machine Learning and nonlinear features in wind turbines. **Renewable Energy**, v. 146, p. 316–328, 2020. ISSN 0960-1481. Cit. on pp. 24, 82, 83, 85, 98, 145, 146.
- ARDUINO. **Arduino Platform (accessed: 15.09.2023)**. 2023. <https://www.arduino.cc/>. Cit. on p. 54.
- AUER, E.; RAUH, A.; KERSTEN, J. Experiments-based parameter identification on the GPU for cooperative systems. eng. **Journal of computational and applied mathematics**, Elsevier B.V, v. 371, p. 112657, 2020. ISSN 0377-0427. Cit. on pp. 67, 68.
- AYALA, H. V. H.; HABINEZA, D.; RAKOTONDRABE, M.; SANTOS COELHO, L. dos. Non-linear black-box system identification through coevolutionary algorithms and radial basis function artificial neural networks. eng. **Applied soft computing**, Elsevier B.V, v. 87, p. 105990–12, 2020. ISSN 1568-4946. Cit. on p. 62.
- BAI, W.; GUO, F.; CHEN, L.; HAO, K.; HUANG, B. Variational Bayesian Inference for Robust Identification of PWARX Systems With Time-Varying Time-Delays. eng. **IEEE transactions on cybernetics**, IEEE, United States, v. 53, n. 6, p. 1–11, 2023. ISSN 2168-2267. Cit. on p. 65.
- BAIDARI, I.; HONNIKOLL, N. Bhattacharyya distance based concept drift detection method for evolving data stream. **Expert Systems with Applications**, v. 183, p. 115303, 2021. ISSN 0957-4174. Cit. on p. 69.
- BAO, C.; ZHANG, S. Algorithm-based fault tolerance for discrete wavelet transform implemented on GPUs. **Journal of Systems Architecture**, v. 108, p. 101823, 2020. ISSN 1383-7621. Cit. on p. 63.
- BEHARRELL, M.; HONARY, F. Decoding Solar Wind–Magnetosphere Coupling. **Space Weather**, v. 14, Sept. 2016. Cit. on pp. 28, 68, 69.

- BEHINFARAZ, R.; GHAEMI, S. Identification and Synchronization of Switching Fractional-Order Complex Networks with Time-Varying Delays Based on a Fuzzy Method. eng. **International journal of fuzzy systems**, Springer Berlin Heidelberg, Berlin/Heidelberg, v. 24, n. 5, p. 2203–2214, 2022. Cit. on p. 65.
- BENCHEIKH, F.; HARKAT, M.; KOUADRI, A.; BENSMAIL, A. New reduced kernel PCA for fault detection and diagnosis in cement rotary kiln. **Chemometrics and Intelligent Laboratory Systems**, v. 204, p. 104091, 2020. ISSN 0169-7439. Cit. on pp. 40, 77, 78.
- BENCHMARK, N. **Nonlinear Benchmark (accessed: 10.10.2023)**. 2023. <https://www.nonlinearbenchmark.org/benchmarks>. Cit. on p. 177.
- BHARDWAJ, R.; XIA, Z.; ANANTHANARAYANAN, G.; JIANG, J.; SHU, Y.; KARIANAKIS, N.; HSIEH, K.; BAHL, P.; STOICA, I. Ekya: Continuous Learning of Video Analytics Models on Edge Compute Servers. **19th USENIX Symposium on Networked Systems Design and Implementation**, 2022. Cit. on pp. 71, 77.
- BILLINGS, S. **Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains**. Chichester, UK: Wiley, 2013. Cit. on pp. 22, 23, 28, 47–49, 53, 62, 88, 89, 94, 128.
- BILLINGS, S.; FAKHOURI, S. Identification of systems containing linear dynamic and static nonlinear elements. **Journal of Mechanical Science and Technology**, 18(1), p. 15–26, 1982. Cit. on p. 48.
- BILLINGS, S.; LEONTARITIS, I. Identification of nonlinear systems using parametric estimation techniques. **Proceedings of the IEE Conference on Control and its Application, Warwick, UK**, p. 183–187, 1981. Cit. on p. 48.
- BILLINGS, S.; VOON, W. Structure detection and model validity tests in the identification of non-linear systems. **IEE Proceedings, Pt. D**, 130(4), p. 193–199, 1984. Cit. on p. 48.
- BILLINGS S.A., C. S.; M.J., K. Identification of MIMO non-linear systems using a forward regression orthogonal estimator. **International Journal of Control**, 49(6), p. 2157–2189, 1989. Cit. on p. 50.
- BILLINGS S.A., K. M.; S., C. Identification of non-linear output affine systems using an orthogonal least-squares algorithm. **International Journal of Systems Science**, 19(8), p. 1559–1568, 1988. Cit. on p. 50.
- BOUNOUA, W.; BAKDI, A. Fault detection and diagnosis of nonlinear dynamical processes through correlation dimension and fractal analysis based dynamic kernel PCA. **Chemical Engineering Science**, v. 229, p. 116099, 2021. ISSN 0009-2509. Cit. on p. 78.

- BRETTEL, M.; FRIEDERICHSEN, N.; KELLER, M.; ROSENBERG, N. How virtualization, decentralization and network building change the manufacturing landscape: An Industry 4.0 Perspective. **International Journal of Science, Engineering and Technology**, v. 8, p. 37–44, Aug. 2014. Cit. on p. 54.
- BURGES, C. J. A tutorial on support vector machines for pattern recognition. **Data mining and knowledge discovery**, Springer, v. 2, n. 2, p. 121–167, 1998. Cit. on p. 44.
- CAO, H.; SUN, P.; ZHAO, L. PCA-SVM method with sliding window for online fault diagnosis of a small pressurized water reactor. **Annals of Nuclear Energy**, v. 171, p. 109036, 2022. ISSN 0306-4549. Cit. on pp. 80, 85.
- CARBALLO HERNÁNDEZ, W.; PELCAT, M.; BHATTACHARYYA, S.; GALÁN, R.; BERRY, F. Flydeling: Streamlined Performance Models for Hardware Acceleration of CNNs through System Identification. **ACM Transactions on Modeling and Performance Evaluation of Computing Systems**, v. 8, May 2023. Cit. on p. 68.
- CARRIR, R.; NUNES, T.; TERRA, S. Manutenção Centrada em Confiabilidade e suas perspectivas na Indústria 4.0 / Reliability centred maintenance: perspectives in Industry 4.0. In: Dec. 2020. Cit. on p. 36.
- CERVANTES, J.; GARCIA-LAMONT, F.; RODRÍGUEZ-MAZAHUA, L.; LOPEZ, A. A comprehensive survey on support vector machine classification: Applications, challenges and trends. **Neurocomputing**, v. 408, p. 189–215, 2020. ISSN 0925-2312. Cit. on p. 45.
- CHANG, C.-C.; LIN, C.-J. LIBSVM: A Library for Support Vector Machines. **ACM Trans. Intell. Syst. Technol.**, Association for Computing Machinery, New York, NY, USA, v. 2, n. 3, May 2011. ISSN 2157-6904. Cit. on p. 78.
- CHEN, L.; MA, J.; LIU, Z.; ZHENG, X.; DUI, H. Reliability analysis and maintenance optimization for the cost-based component maintenance priority with Industry 4.0. In: RAM, M.; XING, L. (Eds.). **Reliability Modeling in Industry 4.0**. Elsevier, 2023. (Advances in Reliability Science). P. 1–13. ISBN 978-0-323-99204-6. Cit. on pp. 21, 24, 36.
- CHEN, S.; HONG X. AD HARRIS, C.; SHARKEY, P. Sparse modelling using orthogonal forward regression with press statistic and regularization. **IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics**, 34(2), p. 898–911, 2004. Cit. on pp. 48, 67.
- CHENG, Y.; WANG, J.; DAI, K.; EL DAMATTY, A. Influence of mechanical faults in pitch, brake, controller systems on fragility of offshore wind turbine under aero-hydro loadings. **Ocean Engineering**, v. 285, p. 115386, 2023. ISSN 0029-8018. Cit. on p. 83.

- CHO, S.; CHOI, M.; GAO, Z.; MOAN, T. Fault detection and diagnosis of a blade pitch system in a floating wind turbine based on Kalman filters and artificial neural networks. **Renewable Energy**, v. 169, p. 1–13, 2021. ISSN 0960-1481. Cit. on p. 78.
- CHOUDHARY; ANURAG; GOYAL, D.; LETHA, S. S. Infrared Thermography-Based Fault Diagnosis of Induction Motor Bearings Using Machine Learning. **IEEE Sensors Journal**, v. 21, n. 2, p. 1727–1734, 2021. Cit. on pp. 80, 85.
- CIANG, C. C.; LEE, J. R.; BANG, H. J. Structural health monitoring for a wind turbine system: a review of damage detection methods. **Measurement Science and Technology**, v. 19, 2008. ISSN 122001. Cit. on pp. 26, 176.
- CIMPOESU, E. M.; CIUBOTARU, B. D.; STEFANOIU; DAN. Fault Detection and Diagnosis Using Parameter Estimation with Recursive Least Squares. In: 2013 19th International Conference on Control Systems and Computer Science. 2013. P. 18–23. Cit. on pp. 70, 77.
- CORREA-JULLIAN, C.; COFRE-MARTEL, S.; SAN MARTIN, G.; LOPEZ DROGUETT, E.; NOVAES PIRES LEITE, G. de; COSTA, A. Exploring Quantum Machine Learning and Feature Reduction Techniques for Wind Turbine Pitch Fault Detection. **Energies**, v. 15, n. 8, 2022. ISSN 1996-1073. Cit. on pp. 83, 85, 146.
- D. JUDE, H. a. B. B. G.; MOHAMED, E.; SWATI VIJAY, S. **Intelligent Edge Computing for Cyber Physical Applications**. Elsevier, Feb. 2023. P. 254. ISBN 0323994121. Cit. on p. 55.
- DAI, W.; NISHI, H.; VYATKIN, V.; HUANG, V.; SHI, Y.; GUAN, X. Industrial Edge Computing: Enabling Embedded Intelligence. **IEEE Industrial Electronics Magazine**, v. 13, n. 4, p. 48–56, 2019. Cit. on pp. 24, 55.
- DAO, P. B. A CUSUM-Based Approach for Condition Monitoring and Fault Diagnosis of Wind Turbines. **Energies**, v. 14, n. 11, 2021. ISSN 1996-1073. Cit. on pp. 76, 77.
- DAO, P. B. On Cointegration Analysis for Condition Monitoring and Fault Detection of Wind Turbines Using SCADA Data. **Energies**, v. 16, n. 5, 2023. ISSN 1996-1073. Cit. on p. 76.
- DAO, P. B.; STASZEWSKI, W. J.; BARSZCZ, T.; UHL, T. Condition monitoring and fault detection in wind turbines based on cointegration analysis of SCADA data. **Renewable Energy**, v. 116, p. 107–122, 2018. Real-time monitoring, prognosis and resilient control for wind energy systems. ISSN 0960-1481. Cit. on p. 76.
- DE SOUZA, G. F. M.; CAMINADA NETTO, A.; DE ANDRADE MELANI, A. H.; DE CARVALHO MICHALSKI, M. A.; DA SILVA, R. F. Chapter 4 - Analysis of systems' reliability and availability. In: DE SOUZA, G. F. M.; CAMINADA NETTO, A.; DE ANDRADE MELANI, A. H.; DE CARVALHO MICHALSKI, M. A.; DA SILVA, R. F. (Eds.). **Reliability Analysis and Asset Management of Engineering Systems**.

-
- Elsevier, 2022. (Advances in Reliability Science). P. 91–117. ISBN 978-0-12-823521-8. Cit. on pp. 21, 36.
- DIBAJ, A.; GAO, Z.; NEJAD, A. R. Fault detection of offshore wind turbine drivetrains in different environmental conditions through optimal selection of vibration measurements. **Renewable Energy**, v. 203, p. 161–176, 2023. ISSN 0960-1481. Cit. on pp. 25, 175.
- DU, S.; WAN, Y.; ZHANG, C.; ZHANG, S. Anomaly Root Cause Analysis for Wind Turbines Based on Denoising Autoencoder and Sparse Estimation. In: 2023 IEEE 12th Data Driven Control and Learning Systems Conference (DDCLS). 2023. P. 449–454. Cit. on pp. 75, 77.
- ELWELL, R.; POLIKAR, R. Incremental learning of concept drift in nonstationary environments. **IEEE Transactions on Neural Networks**, IEEE, v. 22, n. 10, p. 1517–1531, 2011. Cit. on p. 23.
- EPIPHANY. **Epiphany-V: A 1024 processor 64-bit RISC System-On-Chip**. 2016. Available from: <<https://arxiv.org/abs/1610.01832>>. Cit. on p. 55.
- FENG, Z.; QIN, S.; LIANG, M. Time–frequency analysis based on Vold-Kalman filter and higher order energy separation for fault diagnosis of wind turbine planetary gearbox under nonstationary conditions. **Renewable Energy**, v. 85, p. 45–56, 2016. ISSN 0960-1481. Cit. on pp. 83, 176.
- FEURER, M.; HUTTER, F. Hyperparameter optimization. **Automated machine learning: Methods, systems, challenges**, Springer International Publishing, p. 3–33, 2019. Cit. on p. 46.
- FLEISCHER, C.; WAAG, W.; HEYN, H.-M.; SAUER, D. U. On-line adaptive battery impedance parameter and state estimation considering physical principles in reduced order equivalent circuit battery models part 2. Parameter and state estimation. **Journal of Power Sources**, v. 262, p. 457–482, 2014. ISSN 0378-7753. Cit. on pp. 29, 70.
- FLYNN, M. Some computer organizations and their effectiveness. *IEEE Trans Comput* C-21:948. **Computers, IEEE Transactions on**, v. C-21, p. 948–960, Oct. 1972. Cit. on p. 56.
- FORT, A.; LANDI, E.; MUGNAINI, M.; VIGNOLI, V. A Low Complexity Rolling Bearing Diagnosis Technique Based on Machine Learning and Smart Preprocessing. **Sensors**, v. 23, n. 17, 2023. ISSN 1424-8220. Cit. on pp. 73, 77.
- GAJULA, K.; MAREPALLI, L. K.; YAO, X.; HERRERA, L. Recursive Least Squares and Adaptive Kalman Filter-Based State and Parameter Estimation for Series Arc Fault Detection on DC Microgrids. **IEEE Journal of Emerging and Selected Topics in Power Electronics**, v. 10, n. 4, p. 4715–4724, 2022. Cit. on pp. 70, 77.

- GALAR, M.; FERNÁNDEZ, A.; BARRENECHEA, E.; BUSTINCE, H.; HERRERA, F. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. **Pattern Recognition**, v. 44, n. 8, p. 1761–1776, 2011. ISSN 0031-3203. Cit. on p. 46.
- GANESHAN, S.; ELUMALAI, N. K.; ACHAR, R.; LEE, W. K. GVF: GPU-Based Vector Fitting for Modeling of Multiport Tabulated Data Networks. **IEEE Transactions on Components, Packaging and Manufacturing Technology**, v. 10, n. 8, p. 1375–1387, 2020. Cit. on pp. 67, 68.
- GAO, Y.; LIU, S.; LI, F.; LIU, Z. Fault detection and diagnosis method for cooling dehumidifier based on LS-SVM NARX model. **International Journal of Refrigeration**, v. 61, Sept. 2015. DOI: [10.1016/j.ijrefrig.2015.08.020](https://doi.org/10.1016/j.ijrefrig.2015.08.020). Cit. on pp. 81, 82.
- GAO, Z.; LIU, X. An Overview on Fault Diagnosis, Prognosis and Resilient Control for Wind Turbine Systems. **Processes**, v. 9, n. 2, 2021. ISSN 2227-9717. Cit. on pp. 25, 175.
- GAROUDJA, E.; HARROU, F.; SUN, Y.; KARA, K.; CHOUDER, A.; SILVESTRE, S. Statistical fault detection in photovoltaic systems. **Solar Energy**, v. 150, p. 485–499, 2017. ISSN 0038-092X. Cit. on pp. 75, 77.
- GAY, W. **Raspberry Pi Hardware Reference**. Jan. 2014. ISBN 978-1-4842-0800-7. Cit. on p. 54.
- GIUBILATO, R.; CHIODINI, S.; PERTILE, M.; DEBEI, S. An evaluation of ROS-compatible stereo visual SLAM methods on a nVidia Jetson TX2. **Measurement**, v. 140, p. 161–170, 2019. ISSN 0263-2241. Cit. on p. 62.
- GLOBAL Wind Energy Council (GWEC) - Global Wind Report 2023. <https://gwec.net/globalwindreport2023/>. Accessed: 2023-09-14. Cit. on p. 175.
- GOOGLE. **Interest in Edge Computing over time (accessed: 25.10.2019)**. 2023. <https://trends.google.com/trends>. Cit. on p. 55.
- GUANGFEI, Z.; WEN, T.; DA, Z. Ice Detection for Wind Turbine Blades Based on PSO-SVM Method. **Journal of Physics: Conference Series**, IOP Publishing, v. 1087, n. 2, p. 022036, Sept. 2018. Cit. on pp. 80, 83, 85.
- GUO, X.-G.; FAN, X.; AHN, C. K. Adaptive Event-Triggered Fault Detection for Interval Type-2 T-S Fuzzy Systems With Sensor Saturation. **IEEE Transactions on Fuzzy Systems**, PP, May 2020. Cit. on pp. 41, 77, 78.
- HARTENSTEIN, R. The von Neumann Syndrome and the CS Education Dilemma. **Reconfigurable Computing: Architectures, Tools and Applications. Lecture Notes in Computer Science, Berlin**, v. 4943, 2008. Cit. on p. 54.
- HELBING, G.; RITTER, M. Power Curve Monitoring with Flexible EWMA Control Charts. In: 2017 International Conference on Promising Electronic Technologies (ICPET). 2017. P. 124–128. Cit. on pp. 29, 75, 77.

- HEVIA-KOCH, P.; KLINGE JACOBSEN, H. Comparing offshore and onshore wind development considering acceptance costs. **Energy Policy**, v. 125, p. 9–19, 2019. ISSN 0301-4215. Cit. on pp. [24](#), [175](#).
- HOFFMAN, J. I. Chapter 31 - Serial Measurements: Time Series, Control Charts, Cusums. In: HOFFMAN, J. I. (Ed.). **Basic Biostatistics for Medical and Biomedical Practitioners (Second Edition)**. Second Edition: Academic Press, 2019. P. 561–574. ISBN 978-0-12-817084-7. Cit. on p. [42](#).
- HOSSEINABADY, M.; ZAINOL, M. A. B.; NUNEZ-YANEZ, J. **Heterogeneous FPGA+GPU Embedded Systems: Challenges and Opportunities**. 2019. arXiv: [1901.06331 \[cs.DC\]](#). Cit. on pp. [56](#), [62](#).
- HSU, C.-Y.; LIU, W.-C. Multiple time-series convolutional neural network for fault detection and diagnosis and empirical study in semiconductor manufacturing. **Journal of Intelligent Manufacturing**, v. 32, Mar. 2021. Cit. on p. [78](#).
- HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. **IEEE transactions on Neural Networks**, IEEE, v. 13, n. 2, p. 415–425, 2002. Cit. on pp. [46](#), [47](#).
- HSU, J.-Y.; WANG, Y.-F.; LIN, K.-C.; CHEN, M.-Y.; HSU, J. H.-Y. Wind Turbine Fault Diagnosis and Predictive Maintenance Through Statistical Process Control and Machine Learning. **IEEE Access**, v. 8, p. 23427–23439, 2020. Cit. on pp. [75](#), [77](#).
- IDKHAJINE, L.; MONMASSON, E. Embedded Fully FPGA-based Real-Time Simulators for Static Power Converters with Power Switch Characteristics Approximated by Identification. eng. **IEEE transactions on industrial electronics (1982)**, IEEE, p. 1–1, 2021. ISSN 0278-0046. Cit. on p. [62](#).
- IGBA, J.; ALEMZADEH, K.; DURUGBO, C.; EIRIKSSON, E. T. Analysing RMS and peak values of vibration signals for condition monitoring of wind turbine gearboxes. **Renewable Energy**, v. 91, p. 90–106, 2016. ISSN 0960-1481. Cit. on p. [176](#).
- MECHANICAL vibration, evaluation of machine vibration by measurements on non-rotating parts: Horizontal axis wind turbines with gearbox. 2015. Cit. on p. [176](#).
- CONDITION monitoring and diagnostics of wind turbines: Monitoring the drivetrain. 2020. Cit. on p. [176](#).
- IZQUIERDO, J.; MÁRQUEZ, A. C.; URIBETXEBARRIA, J.; ERGUIDO, A. On the importance of assessing the operational context impact on maintenance management for life cycle cost of wind energy projects. **Renewable Energy**, v. 153, p. 1100–1110, 2020. ISSN 0960-1481. Cit. on pp. [25](#), [175](#).

- JANSSENS, O.; SLAVKOVIKJ, V.; VERVISCH, B.; STOCKMAN, K.; LOCCUFIER, M.; VERSTOCKT, S.; VAN DE WALLE, R.; VAN HOECKE, S. Convolutional Neural Network Based Fault Detection for Rotating Machinery. **Journal of Sound and Vibration**, v. 377, p. 331–345, 2016. ISSN 0022-460X. Cit. on pp. 41, 77, 78.
- JII, Y.; LIU, J.; LIU, H. An identification algorithm of generalized time-varying systems based on the Taylor series expansion and applied to a pH process. eng. **Journal of process control**, Elsevier Ltd, v. 128, p. 103007, 2023. ISSN 0959-1524. Cit. on p. 65.
- JIA, Q.; DU, W.; ZHANG, Y. Semi-supervised kernel partial least squares fault detection and identification approach with application to HGPWLTP: KPLS fault detection and identification. **Journal of Chemometrics**, v. 30, July 2016. Cit. on pp. 40, 77, 78.
- JIAYANG, L.; W., X.; WU, S.; WAN, L.; XIE, F. Wind turbine fault detection based on deep residual networks. **Expert Systems with Applications**, v. 213, p. 119102, 2023. ISSN 0957-4174. Cit. on p. 42.
- KAUR, B.; DADKHAH, S.; SHOELEH, F.; NETO, E. C. P.; XIONG, P.; IQBAL, S.; LAMONTAGNE, P.; RAY, S.; GHORBANI, A. A. Internet of Things (IoT) security dataset evolution: Challenges and future directions. **Internet of Things**, v. 22, p. 100780, 2023. ISSN 2542-6605. Cit. on p. 24.
- KHAN, N. Processing-Efficient Distributed Adaptive RLS Filtering for Computationally Constrained Platforms. **Wireless Communications and Mobile Computing**, v. 2017, p. 1–7, May 2017. Cit. on p. 103.
- KHANNA, A.; SHARMA, R.; DHINGRA, A.; DHALIWAL, N. Preventive breakdown and fault detection of machine using industrial IOT in maintenance and automation. **Materials Today: Proceedings**, 2023. ISSN 2214-7853. Cit. on p. 22.
- KIRK, D.; HWU, W. **Programming Massively Parallel Processors: A Hands-on Approach**. Elsevier Science, 2012. ISBN 9780123914187. Available from: <<https://books.google.com.br/books?id=E0Uaag8qicUC>>. Cit. on pp. 56–58.
- KOIDE, K.; MIURA, J.; MENEGATTI, E. Monocular person tracking and identification with on-line deep feature selection for person following robots. **Robotics and Autonomous Systems**, v. 124, p. 103348, 2020. Cit. on p. 62.
- LI, B.; DELPHA, C.; DIALLO, D.; MIGAN-DUBOIS, A. Application of Artificial Neural Networks to photovoltaic fault detection and diagnosis: A review. **Renewable and Sustainable Energy Reviews**, v. 138, p. 110512, 2021. ISSN 1364-0321. Cit. on p. 78.
- LI, C. Parallel Implementation of the Recursive Least Square for Hyperspectral Image Compression on GPUs. **KSII Transactions on Internet and Information Systems**, v. 11, p. 3543–3557, July 2017. Cit. on p. 103.

- LI, C.; SU, Y. P. M.; JIANG, T. GPU Parallel Implementation for Real-Time Feature Extraction of Hyperspectral Images. eng. **Applied sciences**, MDPI AG, v. 10, n. 19, p. 6680, 2020. ISSN 2076-3417. Cit. on p. [63](#).
- LI, G.; MA, X.; WANG, X.; YUE, H.; LI, J.; LIU, L.; FENG, X.; XUE, J. Optimizing deep neural networks on intelligent edge accelerators via flexible-rate filter pruning. **Journal of Systems Architecture**, v. 124, p. 102431, 2022. ISSN 1383-7621. Cit. on p. [62](#).
- LI HUA-LIANG WEI, S. A. B. Identification of nonlinear time-varying systems using an online sliding-window and common model structure selection (CMSS) approach with applications to EEG. **International Journal of Systems Science - Sarrigiannis (2016)**, 47:11, p. 2671–2681, 2016. Cit. on pp. [28](#), [67](#), [70](#), [71](#), [77](#).
- LI, Y.; ZHANG, J.-B.; CUI, W.-G.; YUAN, H.; WEI, H.-L. A Multiple Beta Wavelet-Based Locally Regularized Ultraorthogonal Forward Regression Algorithm for Time-Varying System Identification With Applications to EEG. **IEEE Transactions on Instrumentation and Measurement**, v. 69, n. 3, p. 916–928, 2020a. Cit. on pp. [28](#), [70](#), [77](#).
- LI, Y.; ZHANG, J.-B.; CUI, W.-G.; YUAN, H.; WEI, H.-L. A Multiple Beta Wavelet-Based Locally Regularized Ultraorthogonal Forward Regression Algorithm for Time-Varying System Identification With Applications to EEG. eng. **IEEE transactions on instrumentation and measurement**, IEEE, v. 69, n. 3, p. 916–928, 2020b. ISSN 0018-9456. Cit. on p. [62](#).
- LI, Z.; ZHANG, Y.; DING, A.; ZHOU, H.; LIU, C. Efficient algorithms for task mapping on heterogeneous CPU/GPU platforms for fast completion time. **Journal of Systems Architecture**, v. 114, p. 101936, 2021. ISSN 1383-7621. Cit. on p. [63](#).
- LIU, J.; YANG, G.; LI, X.; HAO, S.; GUAN, Y.; LI, Y. A deep generative model based on CNN-CVAE for wind turbine condition monitoring. **Measurement Science and Technology**, v. 34, Dec. 2022. Cit. on pp. [75](#), [77](#).
- LIU, Y.; WU, Z.; WANG, X. Research on Fault Diagnosis of Wind Turbine Based on SCADA Data. **IEEE Access**, v. 8, p. 185557–185569, 2020. Cit. on pp. [75](#), [77](#).
- LJUNG, L. Perspectives on system identification. **Annual Reviews in Control**, 34, n 1, p. 112, 2010. Cit. on p. [47](#).
- LU, S.; HE, Q.; YUAN, T.; KONG, F. Online Fault Diagnosis of Motor Bearing via Stochastic-Resonance-Based Adaptive Filter in an Embedded System. **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, v. 47, n. 7, p. 1111–1122, 2017. Cit. on pp. [73](#), [77](#).
- LUCKE, E. Chapter 8 - Operation, maintenance, and turnarounds. In: NOURI, M.; LUCKE, E. (Eds.). **Life Cycle of a Process Plant**. Elsevier, 2022. P. 157–186. ISBN 978-0-12-813598-3. Cit. on p. [21](#).

- MAIER, A.; SHARP, A.; VAGAPOV, Y. Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things. In: 2017 Internet Technologies and Applications (ITA). 2017. P. 143–148. Cit. on p. 54.
- MANSOURI, M.; DHIBI, K.; HAJJI, M.; BOUZRARA, K.; NOUNOU, H.; NOUNOU, M. Interval-Valued Reduced RNN for Fault Detection and Diagnosis for Wind Energy Conversion Systems. **IEEE Sensors Journal**, v. 22, p. 1, May 2022. Cit. on p. 80.
- MANSOURI, M.; HARKAT, M.-F.; NOUNOU, H.; NOUNOU, M. **Data-Driven and Model-Based Methods for Fault Detection and Diagnosis**. Feb. 2020. ISBN 9780128191644. Cit. on pp. 39, 40, 44, 78.
- MARWEDEL, P. **Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems**. Springer International Publishing, Jan. 2010. P. 1–27. ISBN 9789400702561. Cit. on pp. 24, 25, 54.
- MCKINNON, C.; TURNBULL, A.; KOUKOURA, S.; CARROLL, J.; MCDONALD, A. Effect of Time History on Normal Behaviour Modelling Using SCADA Data to Predict Wind Turbine Failures. **Energies**, v. 13, p. 4745, Sept. 2020. Cit. on p. 83.
- MELLIT, A.; BENGHANEM, M.; KALOGIROU, S.; MASSI PAVAN, A. An embedded system for remote monitoring and fault diagnosis of photovoltaic arrays using machine learning and the internet of things. **Renewable Energy**, v. 208, p. 399–408, 2023. ISSN 0960-1481. Cit. on pp. 73, 77.
- MITTAL, S. A Survey on optimized implementation of deep learning models on the NVIDIA Jetson platform. **Journal of Systems Architecture**, v. 97, p. 428–442, 2019. Cit. on pp. 59, 62, 63.
- MONESS, M.; MAHMOUD, M. O.; MOUSTAFA, A. M. A Real-Time Heterogeneous Emulator of a High-Fidelity Utility-Scale Variable-Speed Variable-Pitch Wind Turbine. **IEEE Transactions on Industrial Informatics**, v. 14, n. 2, p. 437–447, 2018. Cit. on p. 84.
- MONTGOMERY, D. C. **Introduction to statistical quality control**. Arizona, US: John Wiley Sons (New York), 2009. Cit. on pp. 29, 42, 43.
- MOUSSA, S.; JEBALI BEN GHORBAL, M.; ARBI ZIANI, J.; SLAMA-BELKHODJA, I. Residual-based Short-Circuit fault detection and isolation in LVDC microgrid. **Sustainable Energy Technologies and Assessments**, v. 54, p. 102803, 2022. ISSN 2213-1388. Cit. on p. 41.
- MUTHUKRISHNAN, S.; PALLEKONDA, A.; SARAVANAN, R.; BALASUBRAMANIAN, M. Fault Detection in the Wind Farm Turbine Using Machine Learning Based On SVM Algorithm. **Journal of Physics: Conference Series**, v. 1964, p. 052015, July 2021. Cit. on p. 83.

- NATARAJAN, R.; P, S. R.; BOSE, S. C.; GURURAJ, H.; FLAMMINI, F.; VELMURUGAN, S. Fault detection and state estimation in robotic automatic control using machine learning. **Array**, v. 19, p. 100298, 2023. ISSN 2590-0056. Cit. on p. 22.
- NOWLAN, F. S.; HEAP, H. F. Reliability-centered maintenance. NTIS, 1978. Cit. on p. 36.
- NUMBA. **Numba: A High Performance Python Compiler (accessed: 25.10.2019)**. 2023. <https://numba.pydata.org/>. Cit. on p. 57.
- NVIDIA. **CUDA C++ Programming Guide (accessed: 25.10.2019)**. 2023a. <https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>. Cit. on pp. 57, 58.
- NVIDIA. **Jetson Nano Developer Kit (accessed: 25.10.2019)**. 2019. <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>. Cit. on pp. 59, 62, 91, 134.
- NVIDIA. **Jetson NVIDIA (accessed: 25.10.2019)**. 2023b. <https://www.nvidia.com/pt-br/autonomous-machines/embedded-systems/>. Cit. on pp. 56, 59.
- ODEN, L. Lessons learned from comparing C-CUDA and Python-Numba for GPU-Computing, p. 216–223, 2020. Cit. on p. 58.
- ODGAARD, P. FDI/FTC wind turbine benchmark modelling. In. Cit. on p. 175.
- OH, Y.; KIM, Y.; NA, K.; YOUN, B. D. A deep transferable motion-adaptive fault detection method for industrial robots using a residual-convolutional neural network. **ISA Transactions**, v. 128, p. 521–534, 2022. ISSN 0019-0578. Cit. on p. 22.
- PARTEL, V.; KAKARLA, S. C.; AMPATZIDIS, Y. Development and evaluation of a low-cost and smart technology for precision weed management utilizing artificial intelligence. **Computers and Electronics in Agriculture**, v. 157, p. 339–350, 2019. Cit. on p. 62.
- PAULA, L. de; SOARES, A.; LIMA, T.; A., G. F.; COELHO, C.; DELBEM, A.; MARTINS, W. Parallel Regressions for Variable Selection using GPU. **Computing**, v. 99, n. 3, p. 219–234, 2016. Cit. on pp. 67, 68.
- PENG, C.; SHANG, R.; HUANG, W.; DENG, M.; FANG, R. Design of Embedded Intelligent Online Monitoring System of OLTC Based on Vibration Analysis. In: ZHENG, L.; SUN, C.; GOH, K.-L. (Eds.). **Proceedings of MEACM 2020**. Cham: Springer International Publishing, 2021. P. 435–446. ISBN 978-3-030-67958-3. Cit. on pp. 82, 85.
- PIVOTO, D. G.; DE ALMEIDA, L. F.; DA ROSA RIGHI, R.; RODRIGUES, J. J.; LUGLI, A. B.; ALBERTI, A. M. Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review. **Journal of Manufacturing Systems**, v. 58, p. 176–192, 2021. ISSN 0278-6125. Cit. on p. 21.

- QIAO, L.; ZHANG, Y.; WANG, Q. Fault detection in wind turbine generators using a meta-learning-based convolutional neural network. **Mechanical Systems and Signal Processing**, v. 200, p. 110528, 2023. ISSN 0888-3270. Cit. on p. [83](#).
- RATO, T. J.; REIS, M. S. Fault detection in the Tennessee Eastman benchmark process using dynamic principal components analysis based on decorrelated residuals (DPCA-DR). **Chemometrics and Intelligent Laboratory Systems**, v. 125, p. 101–108, 2013. ISSN 0169-7439. Cit. on pp. [75](#), [77](#).
- RAZZOLINI, C. F. B. **CUDA-Parttree: Estratégia Paralela em GPU para Alinhamento Múltiplo Heurístico de Milhares de Sequências**. 2019. S. 68. MA thesis – Universidade de Brasília, Brasília. Cit. on p. [56](#).
- SACIE, M.; SANTOS, M.; LÓPEZ, R.; PANDIT, R. Use of State-of-Art Machine Learning Technologies for Forecasting Offshore Wind Speed, Wave and Misalignment to Improve Wind Turbine Performance. **Journal of Marine Science and Engineering**, v. 10, n. 7, 2022. ISSN 2077-1312. Cit. on p. [83](#).
- SAHA, S.; ZHAI, X.; EHSAN, S.; MAJEED, S.; MCDONALD-MAIER, K. RASA: Reliability-Aware Scheduling Approach for FPGA-Based Resilient Embedded Systems in Extreme Environments. **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, v. 52, n. 6, p. 3885–3899, 2022. Cit. on p. [73](#).
- SANDERS, J.; KANDROT, E. **CUDA by Example: An Introduction to General-Purpose GPU Programming**. 1st: Addison-Wesley Professional, 2010. ISBN 0131387685. Cit. on p. [57](#).
- SANTOS, P.; VILLA MONTOYA, L.; REÑONES, A.; BUSTILLO, A.; MAUDES, J. An SVM-Based Solution for Fault Detection in Wind Turbines. **Sensors (Basel, Switzerland)**, v. 15, p. 5627–48, Mar. 2015. Cit. on pp. [41](#), [77](#), [83](#), [85](#), [146](#).
- SHAHZAD, F.; HUANG, Z.; MEMON, W. H. Process Monitoring Using Kernel PCA and Kernel Density Estimation-Based SSGLR Method for Nonlinear Fault Detection. **Applied Sciences**, v. 12, n. 6, 2022. ISSN 2076-3417. Cit. on p. [76](#).
- SHAN, N.; XU, X.; BAO, X.; QIU, S. Fast Fault Diagnosis in Industrial Embedded Systems Based on Compressed Sensing and Deep Kernel Extreme Learning Machines. **Sensors**, v. 22, n. 11, 2022. ISSN 1424-8220. Cit. on pp. [72](#), [77](#).
- SHAOUT, A.; EL-MOUSA, A.; KHALID, M. Specification and Modeling of HW/SW CO-Design for Heterogeneous Embedded Systems, London, U.K. **Lecture Notes in Engineering and Computer Science**, v. 2176, p. 273–278, 2009. Cit. on pp. [25](#), [54](#).
- SHEN, M.; WEN, P.; SONG, B.; LI, Y. An EEG based real-time epilepsy seizure detection approach using discrete wavelet transform and machine learning methods. **Biomedical Signal Processing and Control**, v. 77, p. 103820, 2022. ISSN 1746-8094. Cit. on p. [22](#).

- SHEN, T.; DONG, Y.; HE, D.; YUAN, Y. Online identification of time-varying dynamical systems for industrial robots based on sparse Bayesian learning. eng. **Science China. Technological sciences**, Science China Press, Beijing, v. 65, n. 2, p. 386–395, 2022. ISSN 1674-7321. Cit. on p. [65](#).
- SHI, W.; CAO, J.; ZHANG, Q.; LI, Y.; XU, L. Edge Computing: Vision and Challenges. **IEEE Internet of Things Journal**, v. 3, n. 5, p. 637–646, 2016. Cit. on pp. [24](#), [55](#).
- SIDDIQUE, S.; ISLAM, S.; NEON, E. E.; SABBIR, T.; NAHEEN, I. T.; KHAN, R. Deep Learning-based Bangla Sign Language Detection with an Edge Device. **Intelligent Systems with Applications**, v. 18, p. 200224, 2023. ISSN 2667-3053. Cit. on p. [59](#).
- SILVA P. LUZ E., S. G. e. a. Towards better heartbeat segmentation with deep learning classification. **Scientific Reports**, v. 10, n. 20701, 2020. Cit. on p. [63](#).
- SILVA SANTOS, C. E. da; SAMPAIO, R. C.; SANTOS COELHO, L. dos; BESTARD, G. A.; LLANOS, C. H. Multi-objective adaptive differential evolution for SVM/SVR hyper-parameters selection. **Pattern Recognition**, v. 110, p. 107649, 2021. ISSN 0031-3203. Cit. on pp. [29](#), [34](#), [46](#), [78](#), [80](#), [81](#), [134](#), [138](#).
- SILVIO SIMANI, S. F. **Fault Diagnosis and Sustainable Control of Wind Turbines Robust Data-Driven and Model-Based Strategies**. Elsevier, 2018. Cit. on pp. [22](#), [24](#), [37](#), [39](#), [40](#), [175](#).
- SIMANI, S.; FANTUZZI, C.; PATTON, R. J. Model-based fault diagnosis techniques. In: MODEL-BASED Fault Diagnosis in Dynamic Systems Using Identification Techniques. Springer, 2003. P. 19–60. Cit. on pp. [28](#), [40](#), [41](#), [70](#).
- SMITS, G.; JORDAAN, E. Improved SVM regression using mixtures of kernels. In: PROCEEDINGS of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290). 2002. v. 3, 2785–2790 vol.3. Cit. on p. [45](#).
- SMOLA, A. J.; SCHÖLKOPF, B. **Learning with kernels**. Citeseer, 1998. v. 4. Cit. on p. [45](#).
- SONG, B.; TAN, S.; SHI, H.; ZHAO, B. Fault detection and diagnosis via standardized k nearest neighbor for multimode process. **Journal of the Taiwan Institute of Chemical Engineers**, v. 106, p. 1–8, 2020. ISSN 1876-1070. Cit. on p. [78](#).
- TIAN, Y.; LIU, X. A deep adaptive learning method for rolling bearing fault diagnosis using immunity. **Tsinghua Science and Technology**, v. 24, n. 6, p. 750–762, 2019. Cit. on pp. [71](#), [77](#).
- TOWLIAT, M.; GUO, Z.; CIMINI, L. J.; XIA, X.-G.; SONG, A. Multi-Layered Recursive Least Squares for Time-Varying System Identification. **IEEE Transactions on Signal Processing**, v. 70, p. 2280–2292, 2022. Cit. on p. [65](#).
- TRAN, M.-K.; FOWLER, M. Sensor Fault Detection and Isolation for Degrading Lithium-Ion Batteries in Electric Vehicles Using Parameter Estimation with Recursive Least Squares. **Batteries**, v. 6, n. 1, 2020. ISSN 2313-0105. Cit. on pp. [29](#), [76](#), [77](#).

- TSYMBAL, A. The Problem of Concept Drift: Definitions and Related Work, May 2004. Cit. on p. 23.
- ULLAH, S.; KIM, D.-H. Lightweight Driver Behavior Identification Model with Sparse Learning on In-Vehicle CAN-BUS Sensor Data. **Sensors**, v. 20, n. 18, 2020. ISSN 1424-8220. Available from: <<https://www.mdpi.com/1424-8220/20/18/5030>>. Cit. on p. 63.
- VAPNIK, V. The nature of statistical learning theory 991 springer-verlag. **New York**, v. 914, 1995. Cit. on pp. 29, 44, 78.
- VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; KAVURI, S. N.; YIN, K. A review of process fault detection and diagnosis: Part III: Process history based methods. **Computers and Chemical Engineering**, v. 27, n. 3, p. 327–346, 2003. ISSN 0098-1354. Cit. on p. 40.
- VERSAL, X. **Xilinx Versal Platform (accessed: 15.09.2023)**. 2023. <https://www.xilinx.com/products/boards-and-kits/device-family/nav-versal-acap.html>. Cit. on p. 55.
- WANG, H.; ZHAO, H.; ZHANG, J.; MA, D.; LI, J.; WEI, J. Survey on Unmanned Aerial Vehicle Networks: A Cyber Physical System Perspective. **IEEE Communications Surveys and Tutorials**, PP, p. 1–1, Dec. 2019. Cit. on p. 22.
- WANG, X.; MENG, R.; WANG, G.; LIU, X.; LIU, X.; LU, D. The research on fault diagnosis of rolling bearing based on current signal CNN-SVM. **Measurement Science and Technology**, IOP Publishing, v. 34, n. 12, p. 125021, Sept. 2023. Cit. on pp. 80, 85.
- WANG, Y.; LV, L.; HUA, C.; DONG, D.; LI, L. Time-varying nonlinear parameters identification of high-speed train suspension system based on WMA. eng. **Journal of physics. Conference series**, IOP Publishing, Bristol, v. 2581, n. 1, p. 12004, 2023. ISSN 1742-6588. Cit. on p. 65.
- WEI, H.; BILLINGS, S. Improved parameter estimates for non-linear dynamical models using a boot-strap method. **International Journal of Control**, 82(11), p. 1991–2001, 2009a. Cit. on p. 91.
- WEI, H.-L.; BILLINGS, S. Identification of time-varying systems using multiresolution wavelet models. **International Journal of Systems Science**, v. 33, Jan. 2003. Cit. on p. 49.
- WENYI, L.; ZHENFENG, W.; JIGUANG, H.; GUANGFENG, W. Wind turbine fault diagnosis method based on diagonal spectrum and clustering binary tree SVM. **Renewable Energy**, v. 50, p. 1–6, 2013. ISSN 0960-1481. Cit. on p. 83.

- WOLF, W.; JERRAYA, A.; MARTIN, G. Multiprocessor System-on-Chip (MPSoC) Technology. eng. **IEEE transactions on computer-aided design of integrated circuits and systems**, IEEE, v. 27, n. 10, p. 1701–1713, 2008. ISSN 0278-0070. Cit. on pp. [54](#), [62](#).
- WONGCHAI, A.; JENJETI, D. rao; PRIYADARSINI, A. I.; DEB, N.; BHARDWAJ, A.; TOMAR, P. Farm monitoring and disease prediction by classification based on deep learning architectures in sustainable agriculture. **Ecological Modelling**, v. 474, p. 110167, 2022. ISSN 0304-3800. Cit. on p. [22](#).
- XIAOLIN, Z.; PENG, H.; LI, X.; FEI, Z.; YONGPING, W.; GAO, L. Research on Bearing Fault Diagnosis of Wind Turbine Gearbox Based on 1DCNN-PSO-SVM. **Acesso IEEE**, v. 8, p. 192248–192258, 2020. Cit. on pp. [80](#), [83](#), [85](#), [146](#).
- XU, Y.; NASCIMENTO, N. M. M.; DE SOUSA, P. H. F.; NOGUEIRA, F. G.; TORRICO, B. C.; HAN, T.; JIA, C.; P., P. Multi-sensor edge computing architecture for identification of failures short-circuits in wind turbine generators. **Applied Soft Computing**, v. 101, p. 107053, 2021. ISSN 1568-4946. Cit. on p. [84](#).
- XUE, S.; HOWARD, I. Torsional vibration signal analysis as a diagnostic tool for planetary gear fault detection. **Mechanical Systems and Signal Processing**, v. 100, p. 706–728, 2018. ISSN 0888-3270. Cit. on p. [176](#).
- YANG, L.; HUA-LIANG, W.; STEPHEN, B.; PTOLEMAIOS, S. Identification of nonlinear time-varying systems using an online sliding-window and common model structure selection (CMSS) approach with applications to EEG. **International Journal of Systems Science**, v. 47, p. 1–11, Mar. 2015. Cit. on p. [103](#).
- YANG C. TAVASSOLIAN N., H. W. e. a. A Fast Parameter Identification Framework for Personalized Pharmacokinetics. **Scientific Reports**, v. 9, n. 14143, 2019. Cit. on pp. [67](#), [68](#).
- YAOWEN; TATSIS, K. E.; DERTIMANIS, V. K.; SPIRIDONAKOS, M. D.; CHATZI, E. N. Vibration-based monitoring of a small-scale wind turbine blade under varying climate conditions. Part I: An experimental benchmark. **Structural Control Health Monitoring**, v. 28, 2021. Cit. on pp. [112](#), [114](#), [115](#), [128](#), [134](#), [176–178](#).
- YU, Y.; PENG, M.-j.; WANG, H.; MA, Z.-g.; CHENG, S.-y.; XU, R.-y. A continuous learning monitoring strategy for multi-condition of nuclear power plant. **Annals of Nuclear Energy**, v. 164, p. 108544, 2021. ISSN 0306-4549. Cit. on pp. [71](#), [77](#).
- YU X., S. Y.; HJ, Z. An adaptive shortest-solution guided decimation approach to sparse high-dimensional linear regression. **Scientific Reports**, v. 11, n. 24034, 2021. Cit. on p. [67](#).

- ZHANG, D.; GAO, Z. An Ensemble Approach for Fault Diagnosis via Continuous Learning. In: 2021 IEEE 19th International Conference on Industrial Informatics (INDIN). 2021. P. 1–5. Cit. on pp. [71](#), [77](#).
- ZHANG, J.; HOU, J.; ZHANG, Z. Real-time identification of out-of-control and instability in process parameter for gasification process: Integrated application of control chart and kalman filter. **Energy**, v. 238, p. 121845, 2022. ISSN 0360-5442. Cit. on pp. [43](#), [76](#), [77](#).
- ZHANG, Q. Some Implementation Aspects of Sliding Window Least Squares Algorithms. **IFAC Proceedings Volumes**, v. 33, n. 15, p. 763–768, 2000. 12th IFAC Symposium on System Identification (SYSID 2000), Santa Barbara, CA, USA, 21-23 June 2000. ISSN 1474-6670. Cit. on p. [53](#).
- ZHANG, Y.; YU, K.; LEI, Z.; GE, J.; XU, Y.; LI, Z.; REN, Z.; FENG, K. Integrated intelligent fault diagnosis approach of offshore wind turbine bearing based on information stream fusion and semi-supervised learning. **Expert Systems with Applications**, v. 232, p. 120854, 2023. ISSN 0957-4174. Cit. on p. [83](#).
- ZHOU, Q.; CHEN, W.; SONG, S.; GARDNER, J. R.; WEINBERGER, K. Q.; CHEN, Y. A Reduction of the Elastic Net to Support Vector Machines with an Application to GPU Computing. In: PROCEEDINGS of the Twenty-Ninth AAAI Conference on Artificial Intelligence. Austin, Texas: AAAI Press, 2015. (AAAI'15), p. 3210–3216. Cit. on pp. [67](#), [68](#).
- ZHU, Y.; DONG, Z.; CHENG, Z.; HUANG, X.; DONG, Y.; ZHANG, Z. Neural network extended state-observer for energy system monitoring. **Energy**, v. 263, p. 125736, 2023. ISSN 0360-5442. Cit. on pp. [71](#), [77](#).
- ŽVOKELJ, M.; ZUPAN, S.; PREBIL, I. EEMD-based multiscale ICA method for slewing bearing fault detection and diagnosis. **Journal of Sound and Vibration**, v. 370, p. 394–423, 2016. ISSN 0022-460X. Cit. on pp. [40](#), [77](#), [78](#).

Appendix

APPENDIX A – Resumo Estendido em Língua Portuguesa

Título: Monitoramento Embarcado Baseado em GPU para Detecção e Diagnóstico de Falhas em Sistemas Não Lineares Variantes no Tempo - Um Estudo de Caso de Turbina Eólica

Autor: Marlon Marques Soudré

Orientador: Prof. Dr. Carlos Humeberto Llanos Quintero

Coorientador: Prof. Dr. Helon Vicente Hultmann Ayala

Programa de Pós-Graduação em Sistemas Mecatrônicos

Brasília, 07 de dezembro de 2023

Palavras-chave: Monitoramento de Sistemas Variantes no Tempo. Projeto de Sistemas Embarcados. Detecção e Diagnóstico de Falhas. Falhas em Turbinas Eólicas.

Introdução

O monitoramento de sistemas não lineares e variantes no tempo visando detectar e diagnosticar falhas não é uma tarefa trivial, sobretudo quando aplicado a sistemas embarcados e suas restrições (SILVIO SIMANI, 2018). Apesar dos desafios, o desenvolvimento e popularização de tecnologias como *Internet of Things* (IoT) (GIUSTO et al., 2010), ferramentas de aprendizagem de máquina e o paradigma de computação na borda (Edge Computing) (SHI et al., 2016), tem aumentado o interesse no tema. No entanto, ainda são poucos os trabalhos inseridos nesse cenário, havendo um espaço a ser preenchido no que tange a soluções que sejam viáveis de serem embarcadas. Neste sentido, o presente trabalho propõe contribuições baseadas em GPUs no intuito de contribuir para o preenchimento desta lacuna na literatura. Primeiramente é proposto uma estratégia aplicadas a sistemas embarcados baseados em GPUs para identificação de sistemas não lineares e caixa-preta. Mais precisamente, foi desenvolvido uma estratégia de paralelização do algoritmo Forward Regression Orthogonal Least Squares (FROLS) para seleção de modelos autoregressivos parcimoniosos, lineares e não lineares. Em seguida são apresentadas soluções para detecção e classificação de falhas, ambas mapeadas em GPUs e baseadas na análise dos parâmetros do modelo identificado, implementadas com ferramentas de gráficos de controle (*control chart*) (TRAN; FOWLER, 2020; ZHANG, J.; HOU; ZHANG, Z., 2022) e máquinas de vetores de suporte (SVM) (ARCOS JIMÉNEZ et al., 2020; AGASTHIAN et al., 2019), respectivamente. Por fim, as soluções supracitadas foram reunidas, juntamente com a estratégia de estimação recursiva dos parâmetros do modelo dentro de uma janela móvel (SWRLS), a fim

de se estabelecer um algoritmo de monitoramento para detecção e diagnóstico de falhas (SWRLS_SVM_FROLS). As soluções se mostraram viáveis quando validadas com dados reais de uma pá de turbina eólica submetida a variação de temperatura e a falhas decorrentes do acúmulo de gelo e formação de trincas em diferentes escalas (YAOWEN et al., 2021). Vale ressaltar que este estudo de caso é atual e relevante, combinando as características de um sistema não linear variável no tempo, devido a falhas e fatores ambientais, e atuando no paradigma de computação na borda, essenciais especialmente para instalações offshore. Nesse sentido, o estudo de caso escolhido explora a solução proposta em sua totalidade, mostrando a viabilidade de um monitoramento embarcado baseado em GPU para detecção e diagnóstico de falhas em sistemas não lineares variantes no tempo.

Materiais e Métodos

Nesta tese foi desenvolvida uma solução baseada em GPUs para identificação de sistemas não lineares denominada FROLS_GPU. Basicamente, foi realizado mapeamento do algoritmo FROLS (BILLINGS, 2013) para GPUs a fim de se conseguir melhoras no tempo computacional e consumo energético quando comparado a sua versão serial. Na estratégia adotada, cada bloco da GPU fica responsável pelo cálculo de um termo dentre o conjunto de termos M disponíveis no modelo completo NARX (do inglês, *Nonlinear Autoregressive eXogenous*). Assim, segue-se o cálculo da contribuição individual de cada um dos M termos segundo o critério da taxa de redução de erro (ERR), escolhendo-se o termo mais relevante e repetindo o procedimento até que um limite pré estabelecido ρ seja alcançado.

Para a detecção de falas foi adotada a estratégia de análise da variação dos parâmetros do modelo NARX do sistema monitorado, utilizando a ferramenta de *control chart* baseado no CUSUM e em uma média móvel para filtragem dos dados, mapeado para GPUs e denominado CCGPU.

A fim de se classificar os dados, foi implementado um classificador SVM cujos *kernels* (RBF e polinomial) foram mapeados em GPU. A SVM tem seus hiperparâmetros otimizados via NIOTS e recebe os parâmetros do modelo NARX do sistema como *features*, podendo ou não ser acrescido da informação da temperatura. Neste sentido, foram criados 24 modelos diferindo-se entre o tipo de *kernel*, o número de parâmetros utilizados como *features* e o uso ou não da temperatura, conforme ilustrado na Tabela 10.

Por fim, os módulos supracitados foram incorporados em dois algoritmos de monitoramento: (a) SWRLS_FROLS_GPU, que identifica falhas ocorridas no sistema e o (b) SWRLS_SVM_GPU, que além de identificar a ocorrência de falhas, também as classifica.

Ambos os algoritmos fazem uso da estimação recursiva dos parâmetros com janela móvel (SWRLS), que trata as variações paramétricas sofridas pelo sistema monitorado. Caso haja variação estrutural, como pela ocorrência de falhas, o algoritmo é re-identificado de modo online pelo FROLS_GPU.

Resultados e Discussões

A Figura 49 compara o coeficiente de determinação R^2 , o *speedup* e o *power improvement* (ganho na redução do consumo energético) do módulo de identificação FROLS_GPU quando aplicado aos dados do benchmark de uma pá de turbina eólica (detalhado no Apêndice B). Foram feitos testes variando o número de atrasos e saídas (n_u e n_y) e o número de termos do polinômio simplificado.

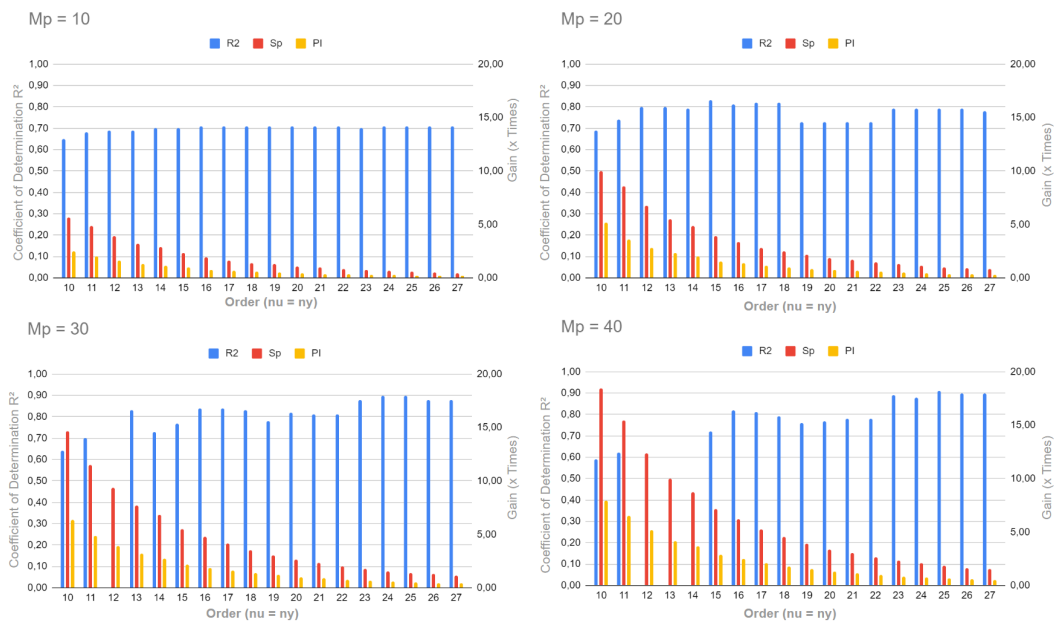


Figure 49 – Relationship between Speedup (red –Sp), Power Improvement (yellow –PI) and Error R^2 (blue – R^2).

O desempenho do algoritmo de identificação de falhas SWRLS_FROLS_GPU (utilizando CCGPU para identificação de falhas) é apresentado na Figura 50 para uma situação que combina variação de temperatura e falhas crescentes de acúmulo de gelo na pá da turbina eólica. São comparados um modelo fixo (sem re-estimação paramétrica - vermelho), um modelo somente com re-estimação paramétrica (preto) e um modelo com re-estimação paramétrica e re-identificação do sistema (azul), sendo verificado que este último apresenta uma aderência maior ao sistema em termos de R^2 .

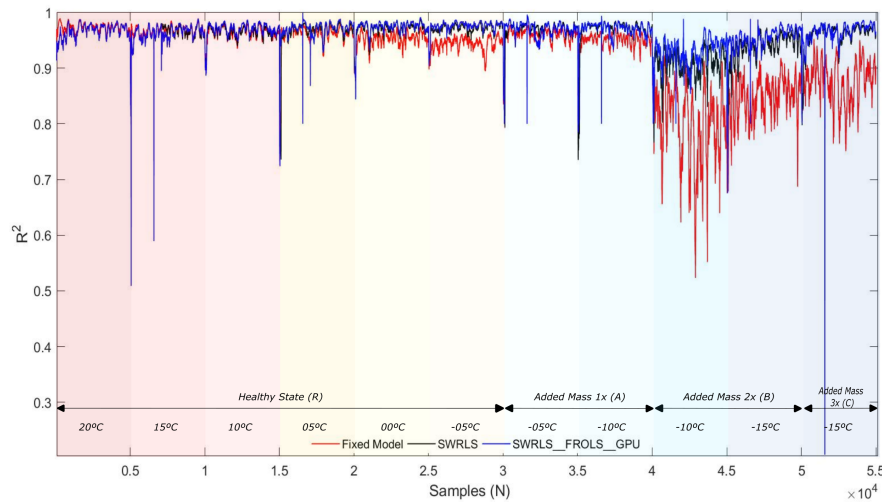


Figure 50 – Error of Wind Turbine System Monitoring - Temperature Variation and Fault.

Por fim, a Figura 51 apresenta o desempenho do algoritmo SWRLS_SVM_GPU (utilizando o classificador SVM_GPU) para detecção e diagnóstico de falhas. Pode ser observado que as classes são classificadas corretamente (curva em verde), assim como os instantes de distúrbio e ocorrência de falha (curva em vermelho).

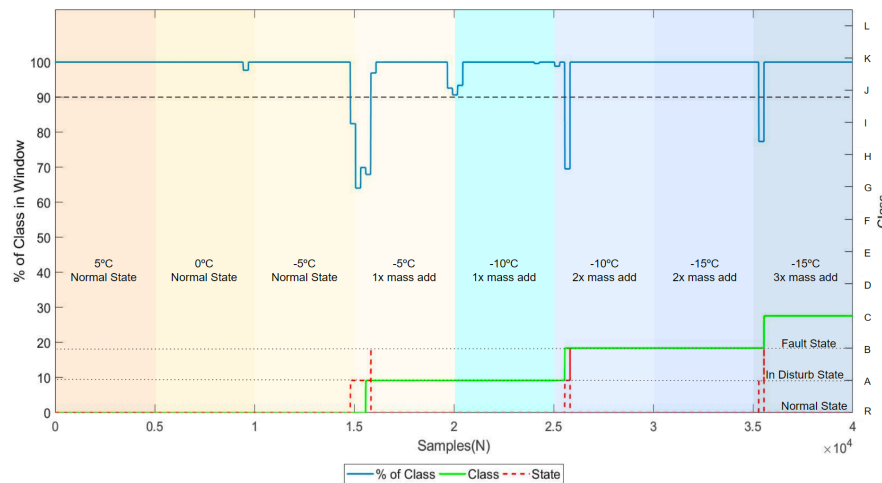


Figure 51 – FDI module performance for the case of wind turbine blade cooling and increasing ice accumulation (RABC_VT case with SVM model composed of 25 parameters and RBF kernel without using temperature).

Conclusão

De forma geral os resultados relatados nesta tese mostram a viabilidade de uma solução embarcada baseada em GPUs para monitoramento de sistemas não lineares variantes no tempo a fim de detectar e diagnosticar falhas destacando os benefícios obtidos

através do mapeamento de GPU em termos de redução de custos computacionais e consumo de energia. Neste sentido, foram propostos módulos para identificação do sistema (FROLS_GPU), detecção de falhas (CCGPU) e classificação de falhas (SVM_GPU), que se mostraram eficazes em termos de desempenho, apresentando um custo computacional e consumo de energia menores quando comparados às soluções tradicionais.

Os algoritmos de detecção de falhas (SWRLS_FROLS_GPU) e de detecção e diagnóstico de falhas (SWRLS_SVM_FROS), compostos com base nos módulos acima mencionados, mostraram-se viáveis quando validados com um conjunto de dados de turbinas eólicas, uma aplicação relevante e atual que combina as características de um sistema não linear e variante no tempo, devido a falhas e fatores ambientais, enquadrando-se no contexto de soluções embarcadas de *edge computing*, especialmente para instalações *offshore*. Nesse sentido, o estudo de caso escolhido explora a solução proposta em sua totalidade, mostrando a viabilidade de um monitoramento embarcado baseado em GPU para detecção e diagnóstico de falhas em sistemas não lineares variantes no tempo.

Referências

- AGASTHIAN, A.; PAMULA, R.; KUMARASWAMIDHAS, A., L. Fault Classification and Detection in Wind Turbine Using Cuckoo-Optimized Support Vector Machine. **Neural Comput. Appl.**, Springer-Verlag, Berlin, Heidelberg, v. 31, n. 5, p. 1503–1511, May 2019. ISSN 0941-0643. Cit. on p. 169.
- ARCOS JIMÉNEZ, A.; ZHANG, L.; GÓMEZ MUÑOZ, C. Q.; GARCÍA MÁRQUEZ, F. P. Maintenance management based on Machine Learning and nonlinear features in wind turbines. **Renewable Energy**, v. 146, p. 316–328, 2020. ISSN 0960-1481. Cit. on p. 169.
- BILLINGS, S. **Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains**. Chichester, UK: Wiley, 2013. Cit. on p. 170.
- GIUSTO, D.; IERA, A.; MORABITO, G.; ATZORI, L. **The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications**. Jan. 2010. ISBN 978-1-4419-1673-0. Cit. on p. 169.
- SHI, W.; CAO, J.; ZHANG, Q.; LI, Y.; XU, L. Edge Computing: Vision and Challenges. **IEEE Internet of Things Journal**, v. 3, n. 5, p. 637–646, 2016. Cit. on p. 169.
- SILVIO SIMANI, S. F. **Fault Diagnosis and Sustainable Control of Wind Turbines Robust Data-Driven and Model-Based Strategies**. Elsevier, 2018. Cit. on p. 169.

- TRAN, M.-K.; FOWLER, M. Sensor Fault Detection and Isolation for Degrading Lithium-Ion Batteries in Electric Vehicles Using Parameter Estimation with Recursive Least Squares. **Batteries**, v. 6, n. 1, 2020. ISSN 2313-0105. Cit. on p. [169](#).
- YAOWEN; TATSIS, K. E.; DERTIMANIS, V. K.; SPIRIDONAKOS, M. D.; CHATZI, E. N. Vibration-based monitoring of a small-scale wind turbine blade under varying climate conditions. Part I: An experimental benchmark. **Structural Control Health Monitoring**, v. 28, 2021. Cit. on p. [170](#).
- ZHANG, J.; HOU, J.; ZHANG, Z. Real-time identification of out-of-control and instability in process parameter for gasification process: Integrated application of control chart and kalman filter. **Energy**, v. 238, p. 121845, 2022. ISSN 0360-5442. Cit. on p. [169](#).

APPENDIX B – Case Studies

The following sections present the case studies used in this work.

B.1 Wind Turbines - A Case Study

Climate change, the limitation of fossil fuels and their environmental impact have led to discussions about viable alternatives for generating energy in a clean and sustainable way, corroborating great progress in the area of renewable energy. Among the most common alternatives are wind turbines (WTs), which are gaining notoriety due to the fact that they are environmentally neutral and have a good cost-effectiveness ratio ([SILVIO SIMANI, 2018](#)). According to the Global Wind Energy Council (GWEC) 2023 report ([GLOBAL..., n.d.](#)), the forecast for wind energy capacity between 2023-2030 is 143 GW, with the 2 TW milestone expected to be reached in just 7 years.

One of the crucial points for making renewable energy sources viable is the economic factor. In this sense, taking into account the cost required to set up and maintain wind farms, there is a concern about the cost-benefit ratio involved, with the desire for these systems to be as efficient and long-lasting as possible. Efficiency must therefore be increased and Operations and Maintenance services (O&M) minimized. By way of analysis, energy costs can increase by 30% due to O&M, especially in offshore installations ([ODGAARD, 2012](#)).

Offshore wind turbines (OWTs) have positive characteristics when compared to onshore turbines, due to access to higher quality wind energy, the possibility of installing larger turbines and the avoidance of operating problems such as noise, visual pollution and other nuisances to people's daily lives ([SILVIO SIMANI, 2018](#)). However, this type of hostile operating environment brings greater vulnerability to the system, increasing the likelihood of unexpected breakdowns and serious damage. In addition, due to the limited access to these structures, their O&M costs are significantly higher than those of onshore systems. According to statistics ([HEVIA-KOCH; KLINGE JACOBSEN, 2019](#)), O&M costs represent approximately 20% to 25% of the total cost of a OWT. That said, it is noteworthy that solutions involving OWTs demand an efficient monitoring system, as they require greater reliability, safety, availability and ease of maintenance than onshore projects ([IZQUIERDO et al., 2020](#); [GAO, Z.; LIU, X., 2021](#); [DIBAJ; GAO, Z.; NEJAD, 2023](#)).

Currently, most turbines have a simple conservative approach to failures, which consists of shutting down the system and waiting for maintenance. Given this fact and the characteristics of offshore installations, a reasonable monitoring alternative is embedded systems, taking into account mainly the isolation of the installations, fitting into the edge

computing paradigm. In this sense, both data acquisition and processing, in the sense of detection and possible fault diagnosis, would be carried out by a dedicated system, providing the advantages inherent in such systems (see subsection 2.3).

It is crucial to point out that blades represent a critical component of wind turbines and failures that occur in these elements produce significant consequences, resulting in substantial losses, especially financial ones. Furthermore, detecting such faults is more challenging compared to other potential problems affecting wind turbine systems (CIANG; LEE; BANG, 2008). In light of these factors, identifying faults in wind turbines is a non-trivial task due to the inherent exposure of such systems to various forms of variation. Thus, the importance of having an adequate monitoring and fault detection system for wind turbines becomes evident, above all for offshore installations with edge computing, justifying the scientific efforts dedicated to solving this problem.

B.1.1 Benchmark Used

There are numerous approaches to data acquisition for monitoring wind turbines, the most common of which are based on vibration and temperature (XUE; HOWARD, 2018; IGBA et al., 2016; FENG; QIN; LIANG, 2016). This is due to the fact that system defects induce rapid changes in vibration signals, especially in relation to the blades, corroborating the fact that monitoring based on such signals is one of the standard methods for detecting this type of fault, and is even recommended by some standards such as ISO 10816-21 (MECHANICAL..., 2015) and ISO 16079-2 (CONDITION..., 2020). In this sense, the present work was developed based on the benchmark built on vibration-based monitoring data from a small-scale wind turbine blade subjected to temperature variation and ice accumulation and cracking failures (YAOWEN et al., 2021).

The benchmark in question was built with the aim of establishing a reference case study for the structural health monitoring (SHM) community:

- Two humidity and temperature sensors, installed on the downwind side of the blade, to ensure that not only the ambient temperature but also that of the blade surface has reached the target value.
- An electromechanical shaker device (*Data Physics SignalForce V4*), for the dynamic excitation of the blade.
- 8 accelerometers (*PCB Piezotronics*), due to the fact that they constitute a conventional and well-established detection technology in the context of SHM applications, suitable for vibration-based global damage detection methods.
- 18 strain gauges, comprising two types of device: unidirectional strain gauges (HBM 1-LY48-20/120) and rosettes (HBM 1-RY88-6/120).

The positioning of the sensors and actuators described above is illustrated in Figure 52, adapted from the benchmark's own article (YAOWEN et al., 2021).

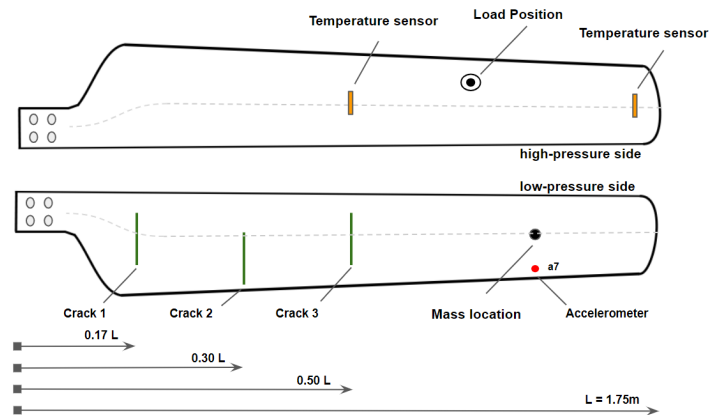


Figure 52 – Sensors configuration on the wind turbine blade (Figure adapted from (YAOWEN et al., 2021)).

This benchmark deals with two types of failure. The first concerns ice buildup, which is a fairly typical phenomenon in structures located in cold climate areas and can affect various aspects of a wind turbine system when it occurs on the blades. Ice formation mainly affects aerodynamic performance, which is significantly disturbed by the increased roughness on the blade surface and leads to reduced energy production or overloading in the system. In addition, all turbine components are exposed to increased loads, induced by the additional mass of ice, while the service life of the structure can be significantly reduced due to the imbalance between the blades (YAOWEN et al., 2021). The second type of failure concerns cracks, of varying length and location, being physically introduced into the structure as surface cuts.

A total of 12 failure situations and one healthy, i.e. failure-free, situation are derived, as shown in Table 14. For each case, input data (electromechanical stirrer) and output data (sensor readings) were taken for 12 different temperatures (from -15°C to 40°C , ranging from 5°C to 5°C). More details on the benchmark in question can be found in the article itself (YAOWEN et al., 2021).

Figure 53 shows the input force and acceleration (output) measurements.

B.2 Electro-Mechanical Positioning System (EMPS)

Electro-Mechanical Positioning System (EMPS) is a standard configuration of a drive system for prismatic joints of robots or machine tools (A. JANOT; BRUNOT, 2019). This benchmark makes up the list of nonlinear dynamical systems benchmarks presented on (BENCHMARK, 2023), a website that brings together the abstracts and presentations that

Table 14 – Table with cases of failures present in the benchmark of wind turbine blades (YAOWEN et al., 2021).

Case Label	Description		
R	Healthy State		
A	Added Mass 1x44g		
B	Added Mass 2x44g		
C	Added Mass 3x44g		
D	Crack 1: $l_1 = 5cm$		
E	Crack 1: $l_1 = 5cm$	Crack 1: $l_1 = 5cm$	
F	Crack 1: $l_1 = 5cm$	Crack 1: $l_1 = 5cm$	Crack 1: $l_1 = 5cm$
G	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 5cm$	Crack 1: $l_1 = 5cm$
H	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 5cm$
I	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 10cm$
J	Crack 1: $l_1 = 15cm$	Crack 1: $l_1 = 10cm$	Crack 1: $l_1 = 10cm$
K	Crack 1: $l_1 = 15cm$	Crack 1: $l_1 = 15cm$	Crack 1: $l_1 = 10cm$
L	Crack 1: $l_1 = 15cm$	Crack 1: $l_1 = 15cm$	Crack 1: $l_1 = 15cm$

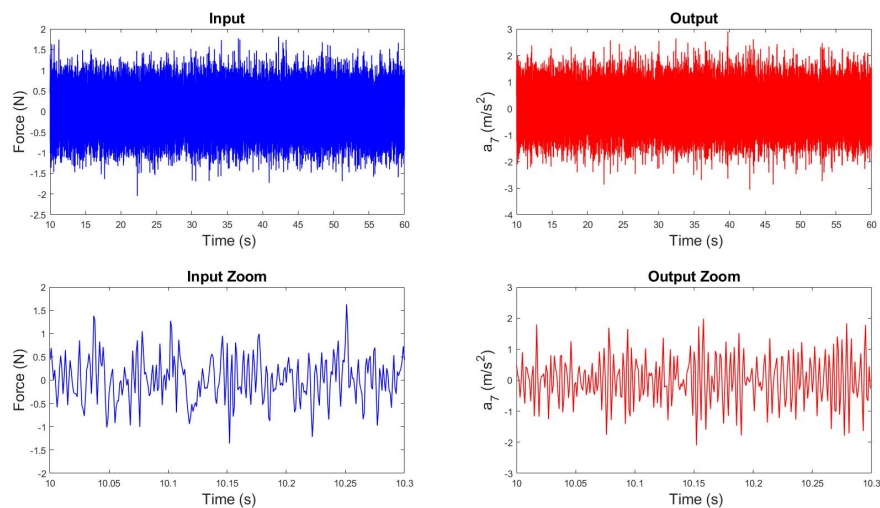


Figure 53 – Input force measurements (blue) and the output acceleration measurements (red) by channel 7 (a_7) for Healthy State (R) at 25°C.

were part of the different editions of the Workshop on Nonlinear System Identification Benchmarks and the results of recent invited sessions related to this theme, being supported by Eindhoven University of Technology and SIEMENS.

It is an interesting data-driven modeling benchmark as it presents nonlinearities related to friction commonly found in robotic manipulators. The EMPS is composed of a DC motor with an encoder, a ball screw drive positioning unit, and a load in translation. This dataset refers to a problem of great industrial application that usually requires continuous monitoring actions for time-varying features, such as load changes and the natural wear of the components. All these mentioned factors imply that the system model presents temporal variations in the structure and related parameters.

Figure 54 shows measurements of force/torque (the *input*) and position (the *output*) of EMPS.

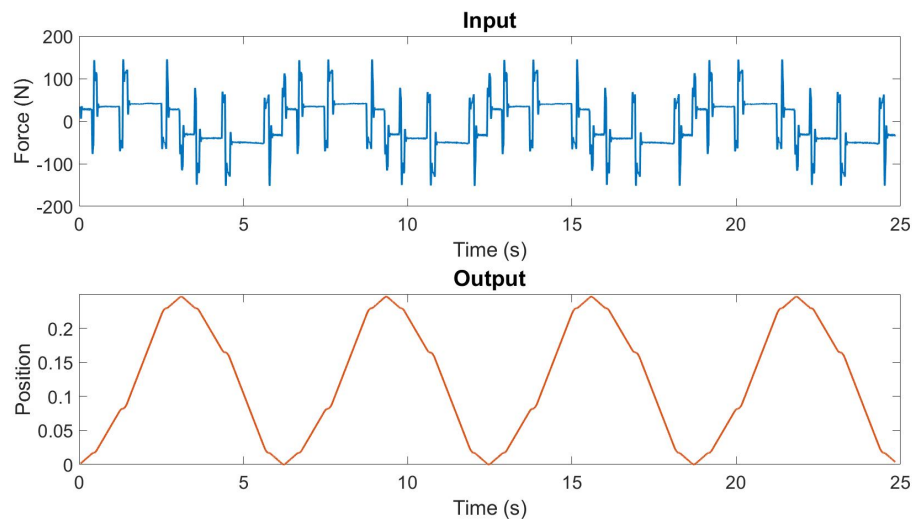


Figure 54 – Force (input) and position (output) data of Electro-Mechanical Positioning System (EMPS).

Annex

ANNEX A – ROC Curve of the SVM_GPU Classifier

The figures below represent the ROC (Receiver Operating Characteristic) curves and the AUC (area under the ROC curve) for each of the models tested with the SVM_GPU classifier.

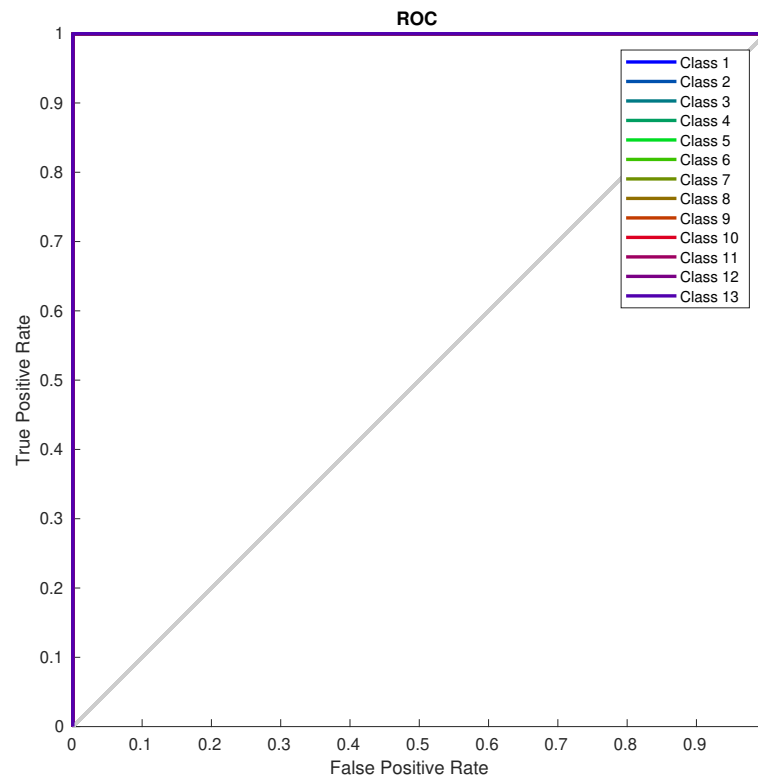


Figure 55 – ROC curve of the SVM_GPU Classifier - model with 10 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

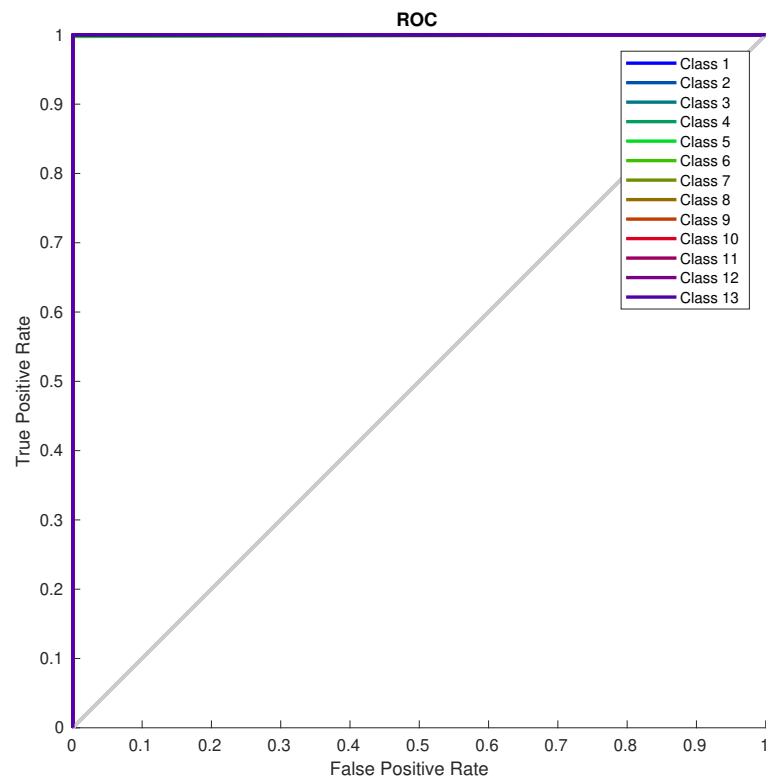


Figure 56 – ROC curve of the SVM_GPU Classifier - model with 10 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 0.9996; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 0.9988; class 6(I) - 1; class 7(L) - 1; class 8(D) - 0.9999; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

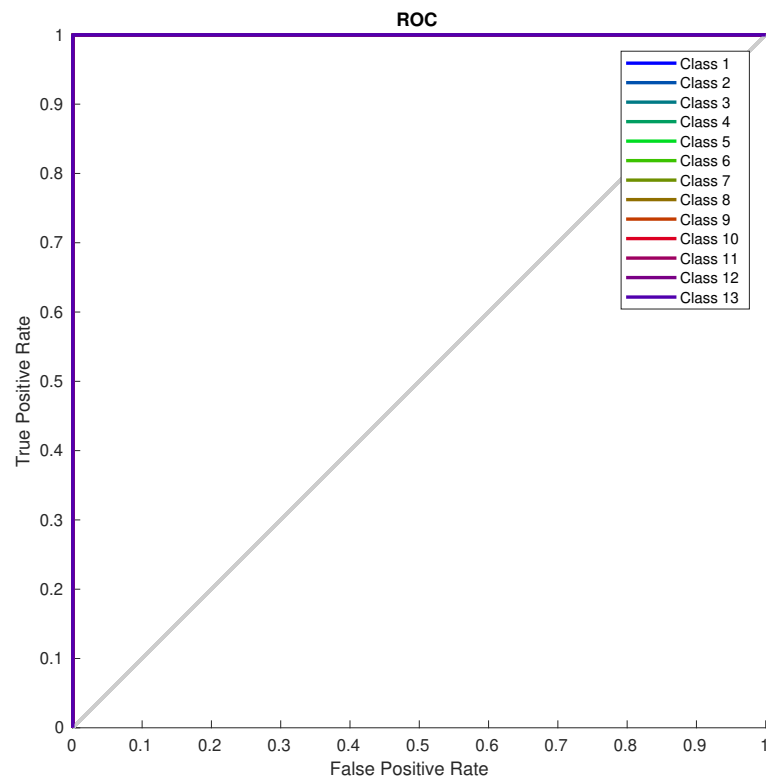


Figure 57 – ROC curve of the SVM_GPU Classifier - model with 10 parameters with temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

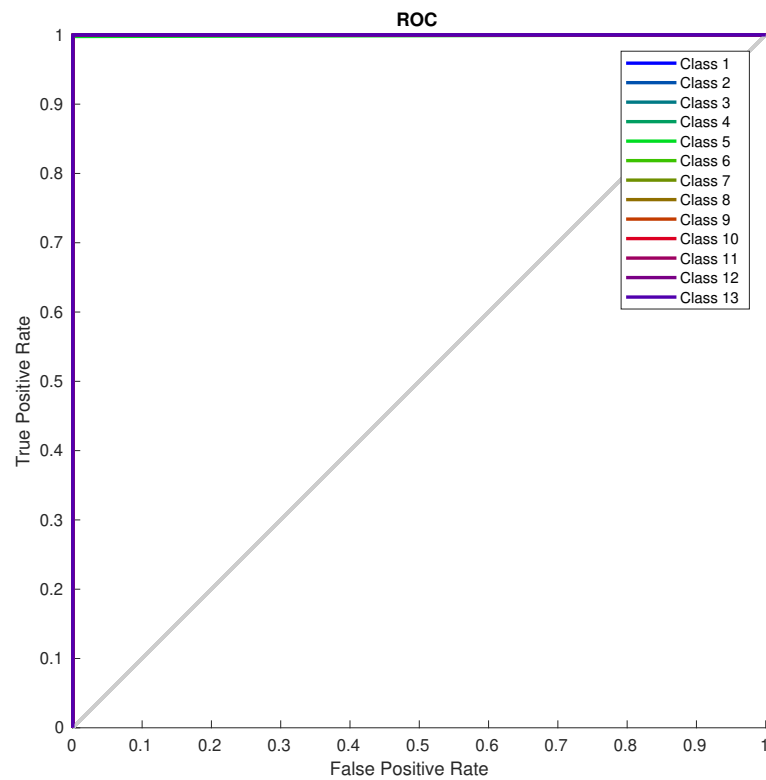


Figure 58 – ROC curve of the SVM_GPU Classifier - model with 10 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 0.99; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 0.99; class 6(I) - 1; class 7(L) - 1; class 8(D) - 0.99; class 9(E) - 0.9941; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

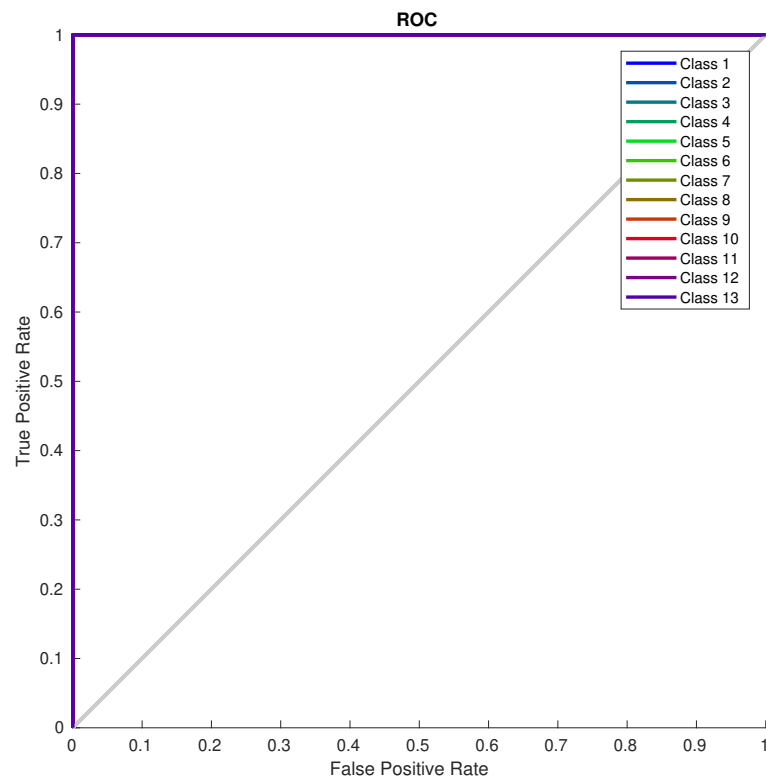


Figure 59 – ROC curve of the SVM_GPU Classifier - model with 15 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

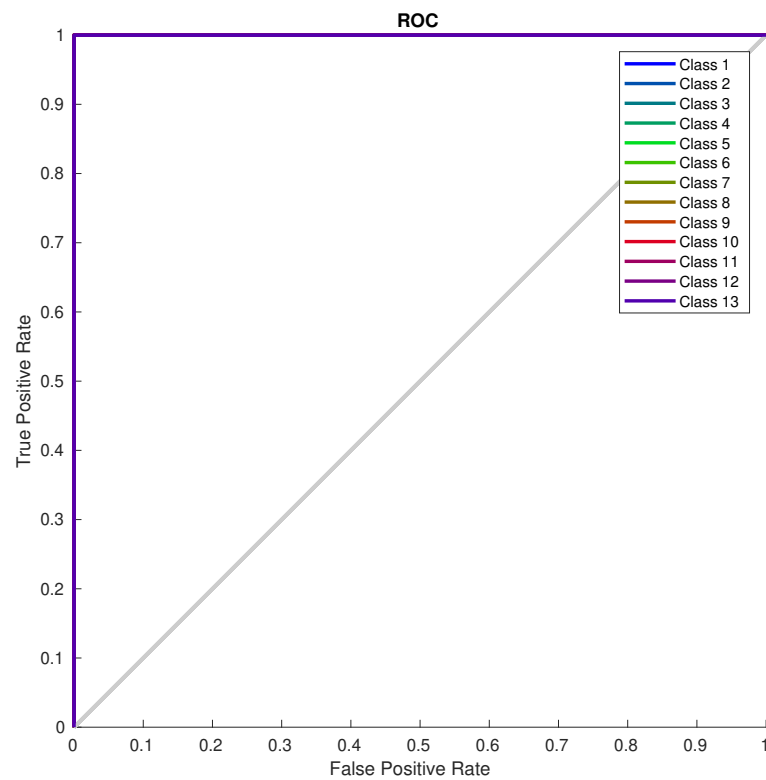


Figure 60 – ROC curve of the SVM_GPU Classifier - model with 15 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

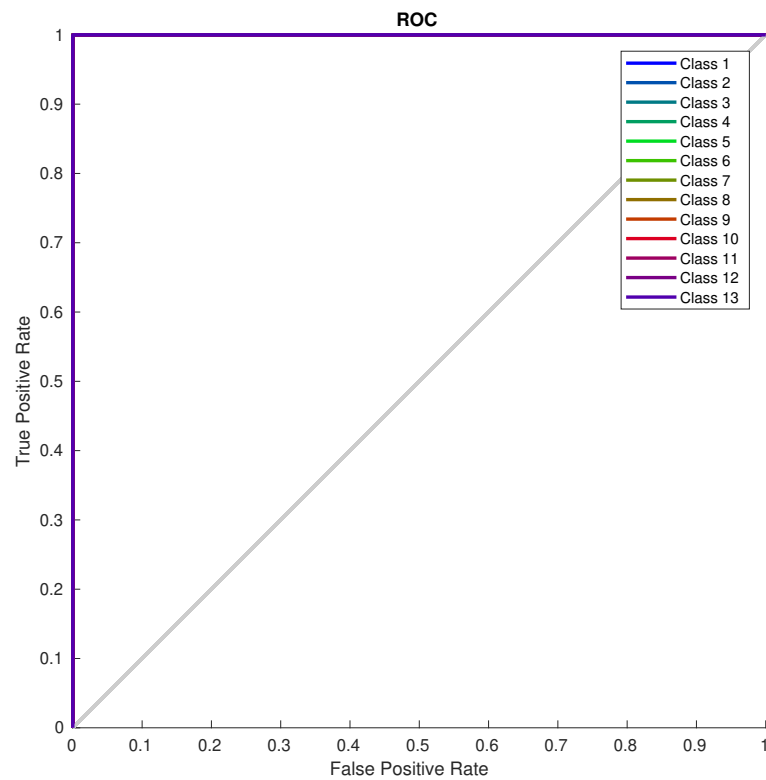


Figure 61 – ROC curve of the SVM_GPU Classifier - model with 15 parameters with temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

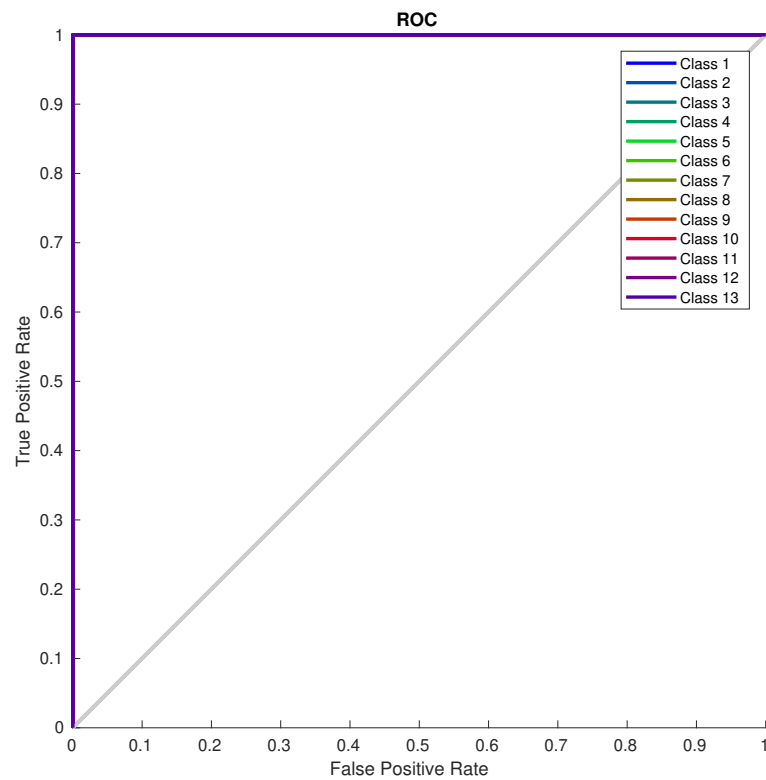


Figure 62 – ROC curve of the SVM_GPU Classifier - model with 15 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

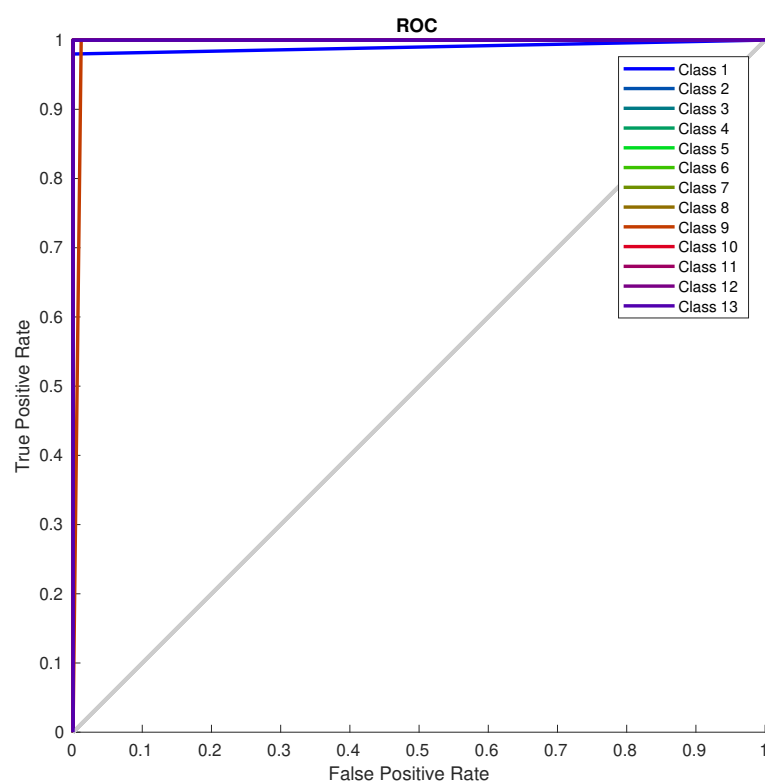


Figure 63 – ROC curve of the SVM_GPU Classifier - model with 20 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 0.99; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 0.9941; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

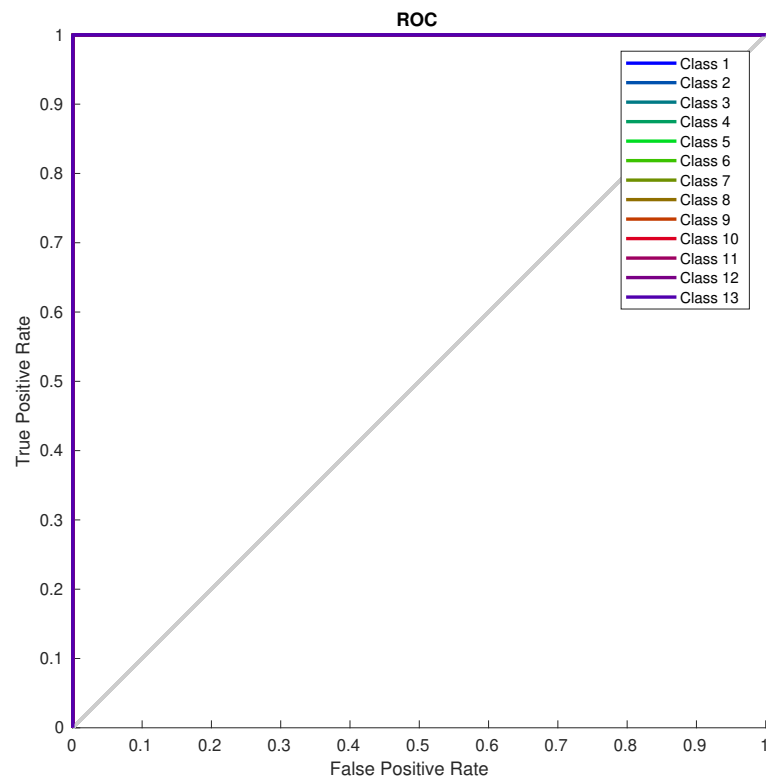


Figure 64 – ROC curve of the SVM_GPU Classifier - model with 20 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

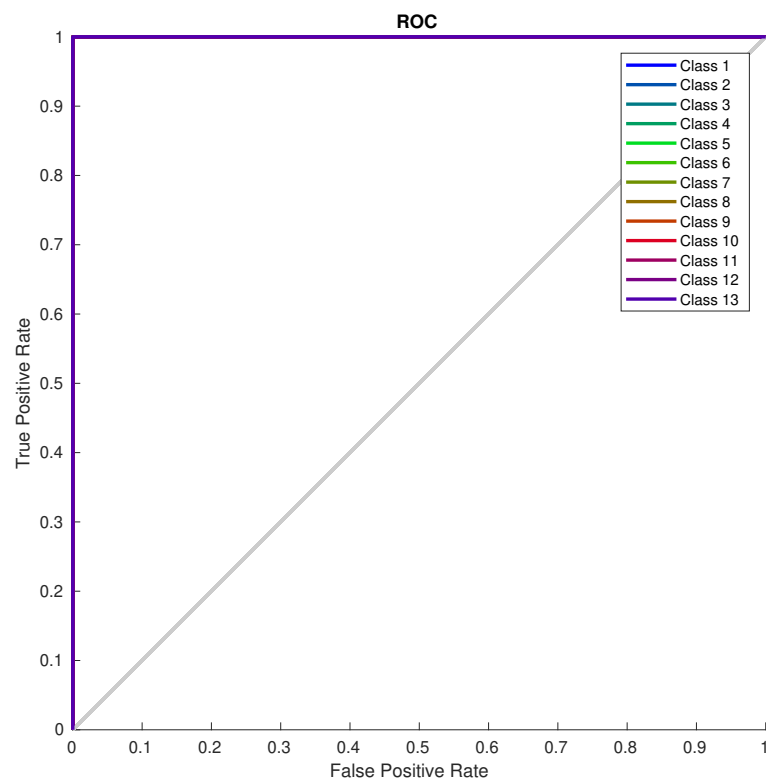


Figure 65 – ROC curve of the SVM_GPU Classifier - model with 20 parameters with the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

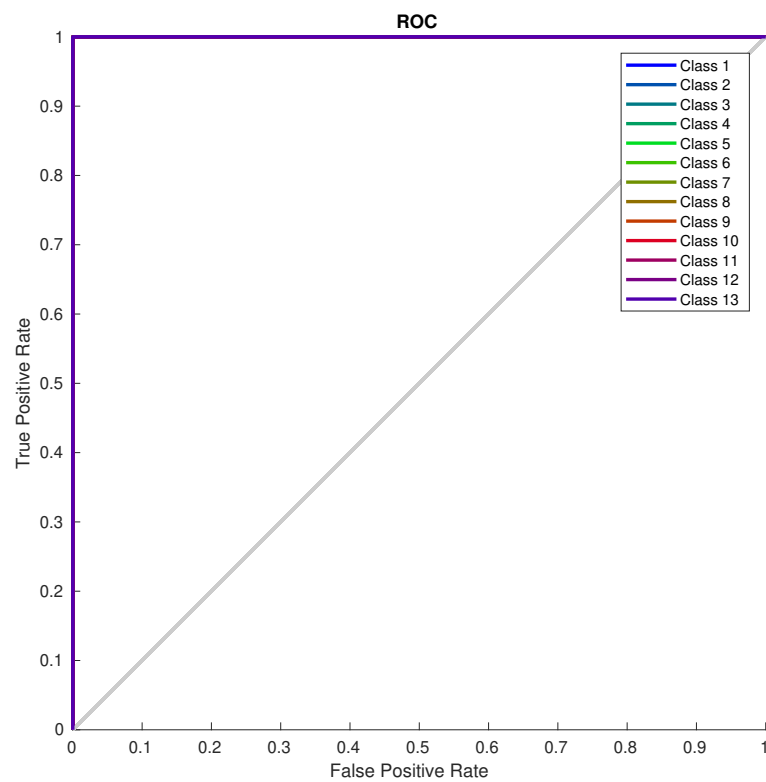


Figure 66 – ROC curve of the SVM_GPU Classifier - model with 20 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

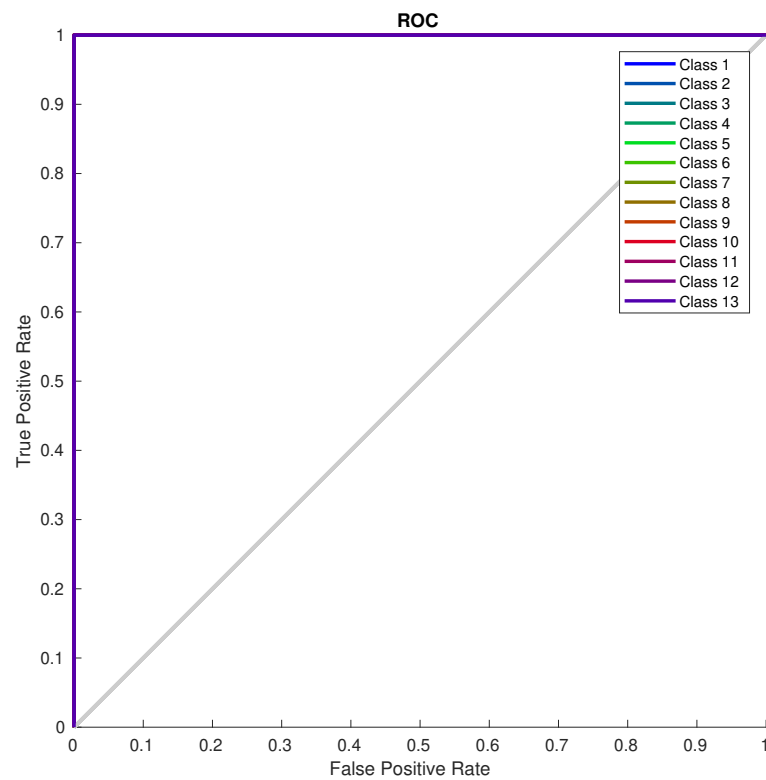


Figure 67 – ROC curve of the SVM_GPU Classifier - model with 25 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

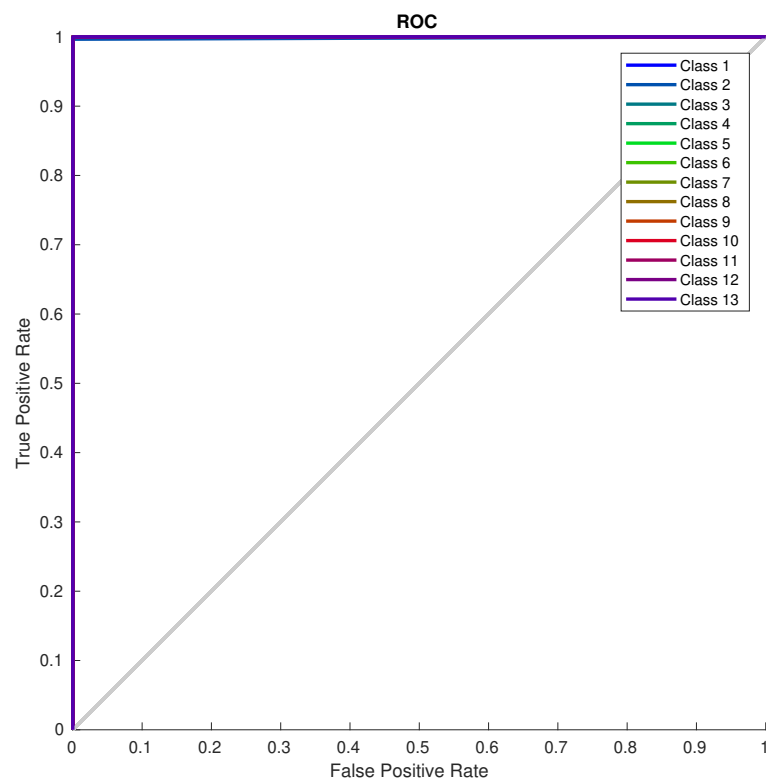


Figure 68 – ROC curve of the SVM_GPU Classifier - model with 25 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 0.9983; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 0.9999;

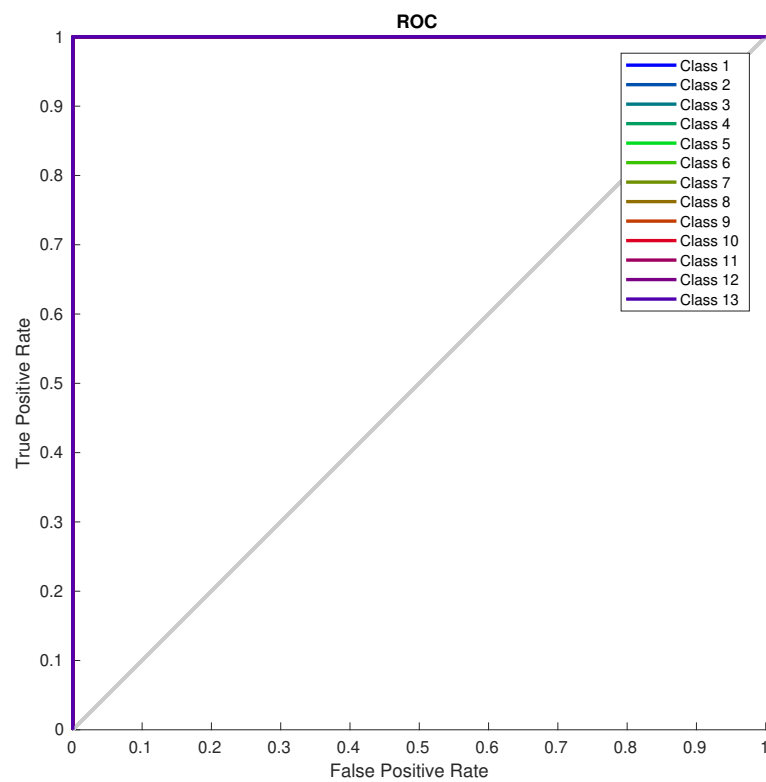


Figure 69 – ROC curve of the SVM_GPU Classifier - model with 25 parameters with the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

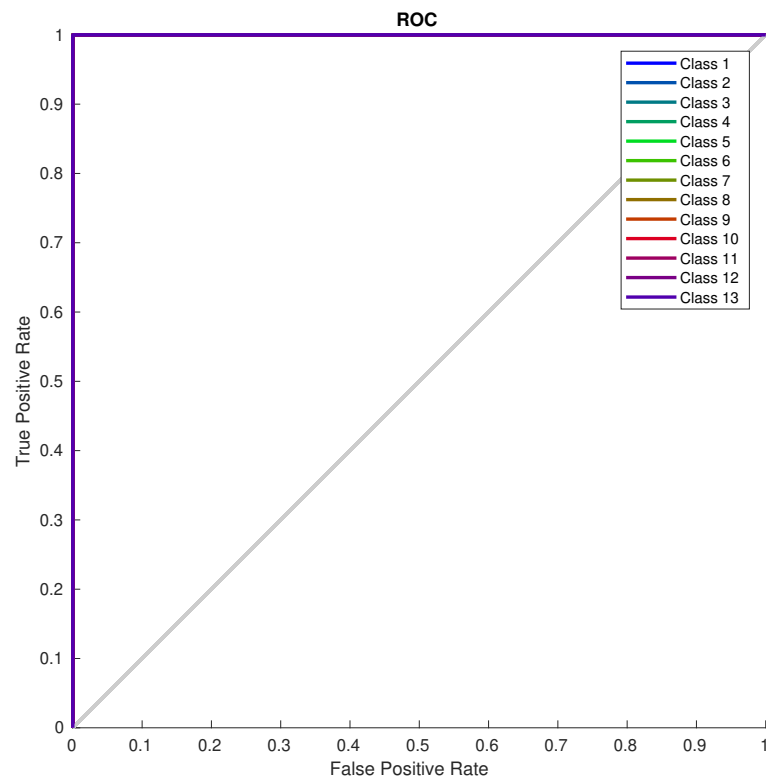


Figure 70 – ROC curve of the SVM_GPU Classifier - model with 25 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

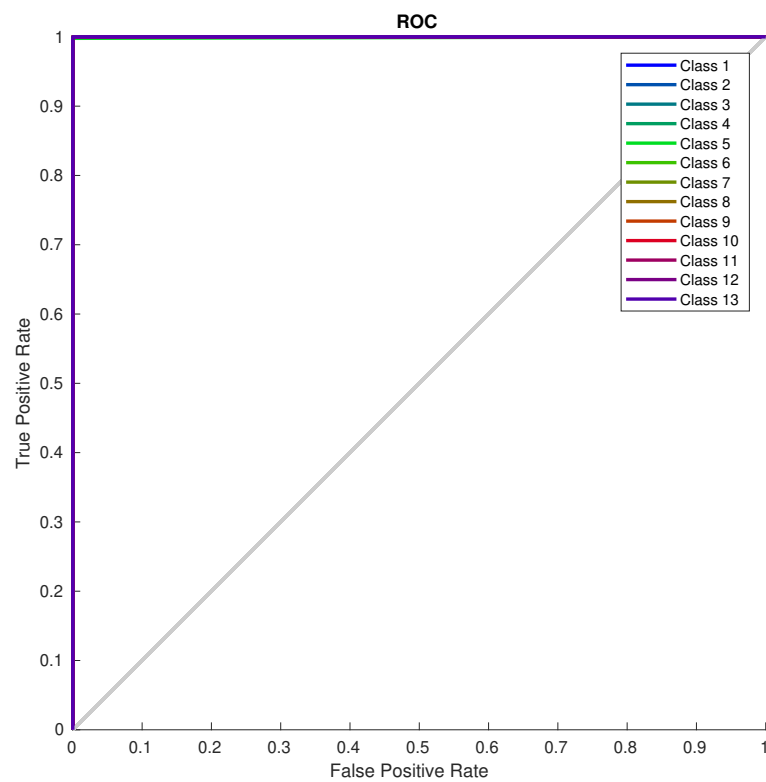


Figure 71 – ROC curve of the SVM_GPU Classifier - model with 30 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 0.9995; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 0.9989; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 0.9999; class 10(G) - 1; class 11(H) - 1; class 12(J) - 0.9999; class 13(K) - 1;

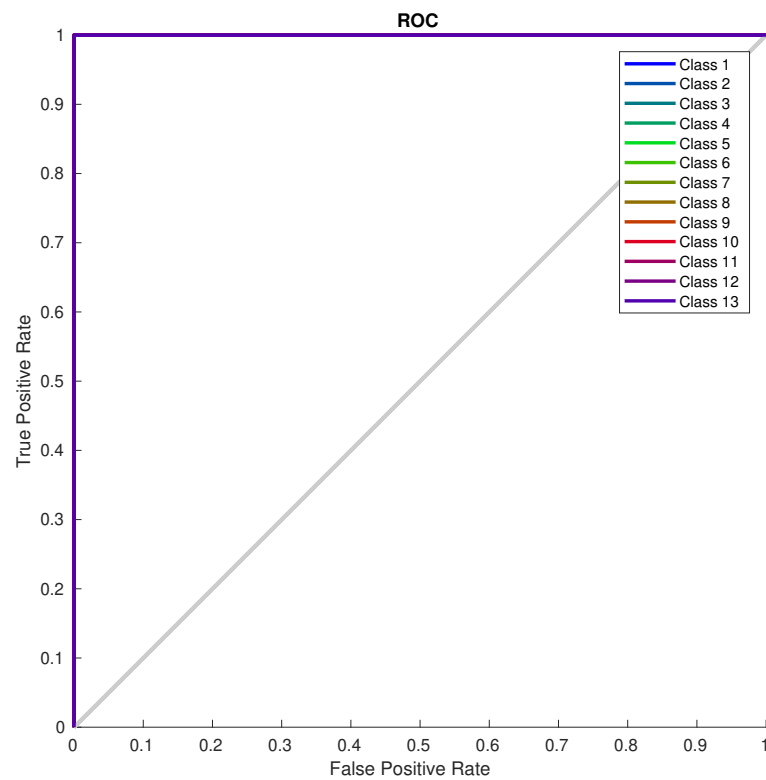


Figure 72 – ROC curve of the SVM_GPU Classifier - model with 30 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 0.9999; class 2(A) - 0.9983; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 0.9999; class 13(K) - 1;

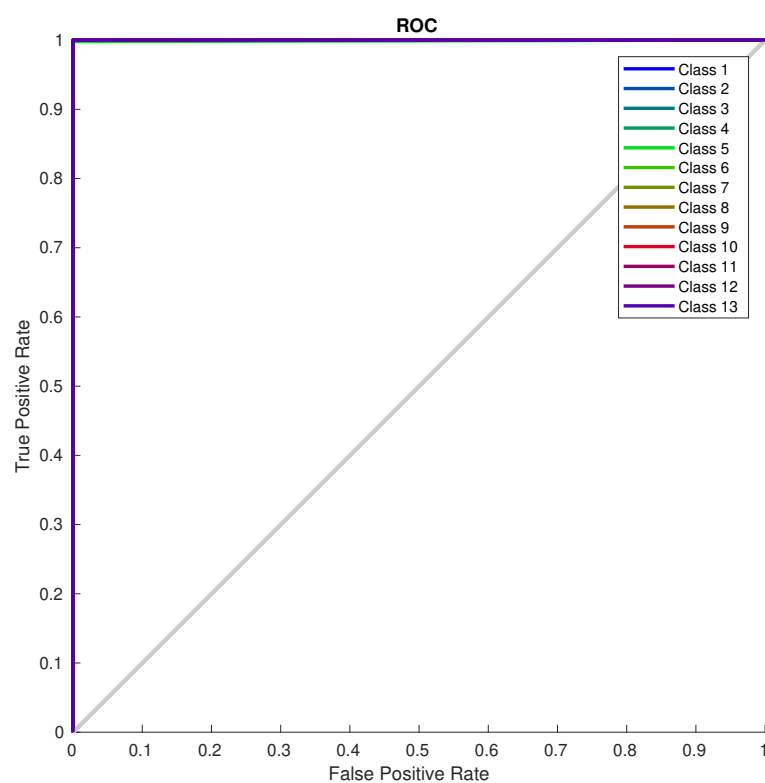


Figure 73 – ROC curve of the SVM_GPU Classifier - model with 30 parameters with the use of temperature and RBF kernel. AUC for class: class 1(R) - 0.9998; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 0.9989; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

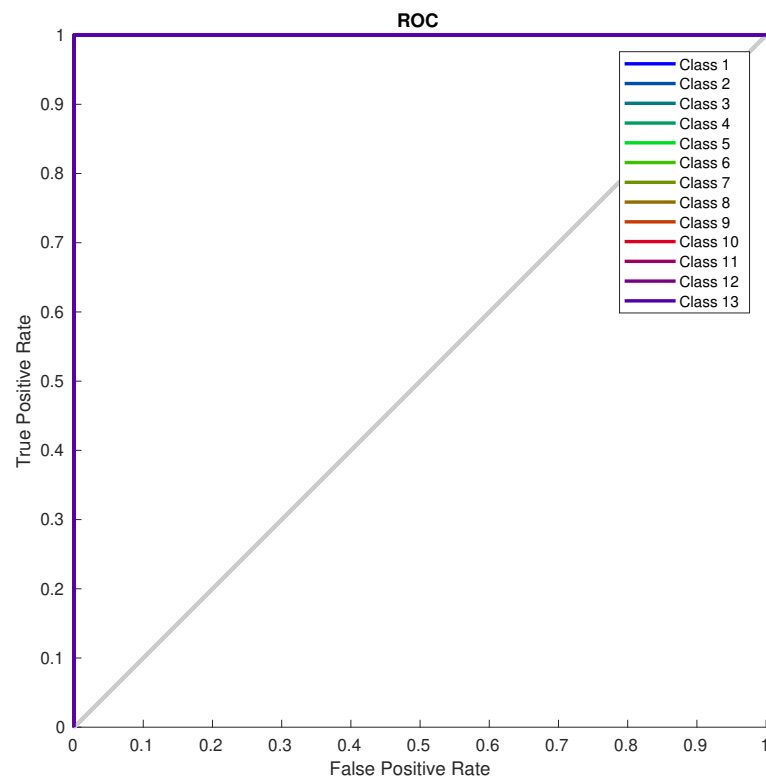


Figure 74 – ROC curve of the SVM_GPU Classifier - model with 30 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

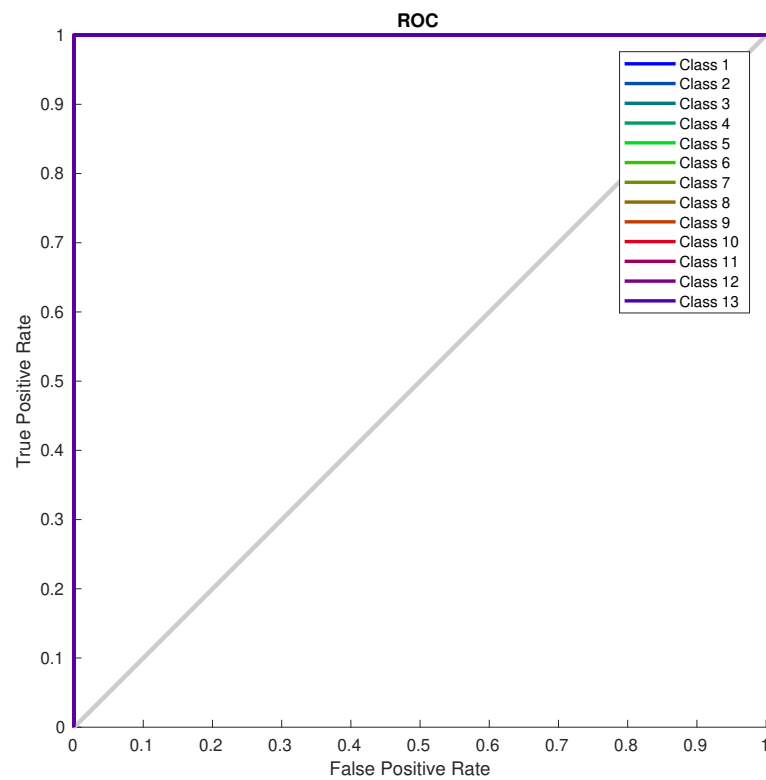


Figure 75 – ROC curve of the SVM_GPU Classifier - model with 35 parameters with temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

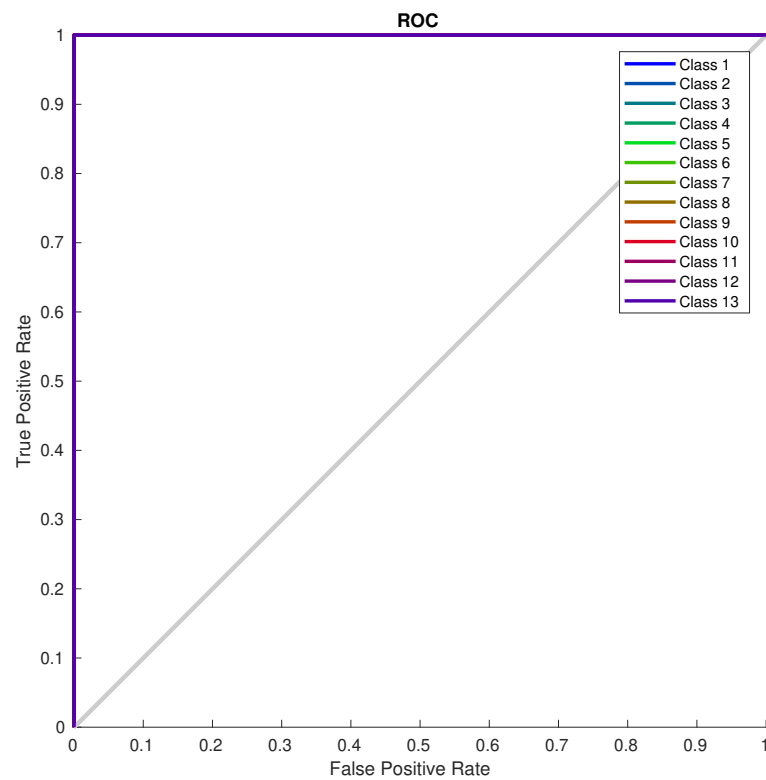


Figure 76 – ROC curve of the SVM_GPU Classifier - model with 35 parameters without the use of temperature and Polynomial kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

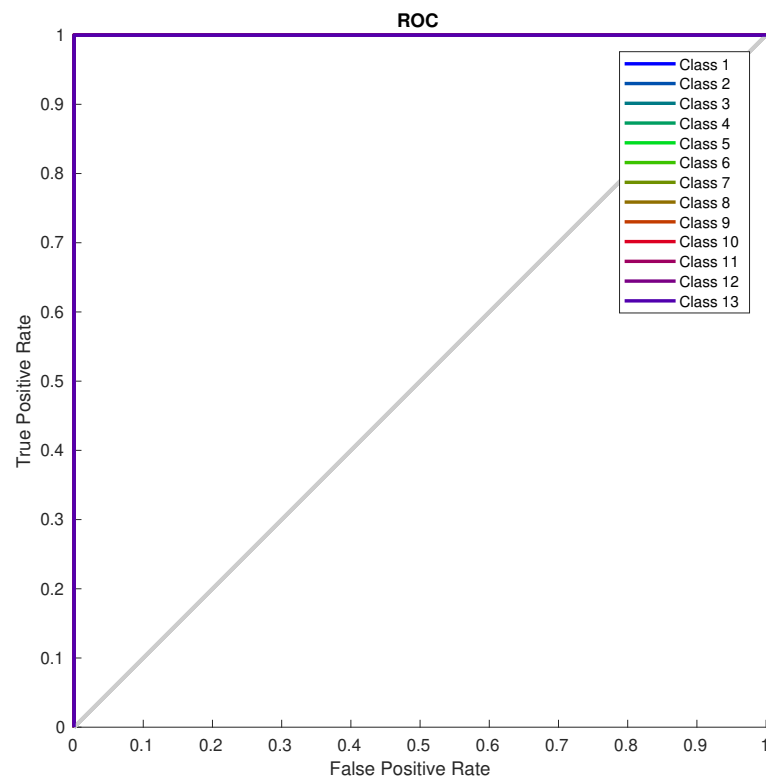


Figure 77 – ROC curve of the SVM_GPU Classifier - model with 35 parameters with the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;

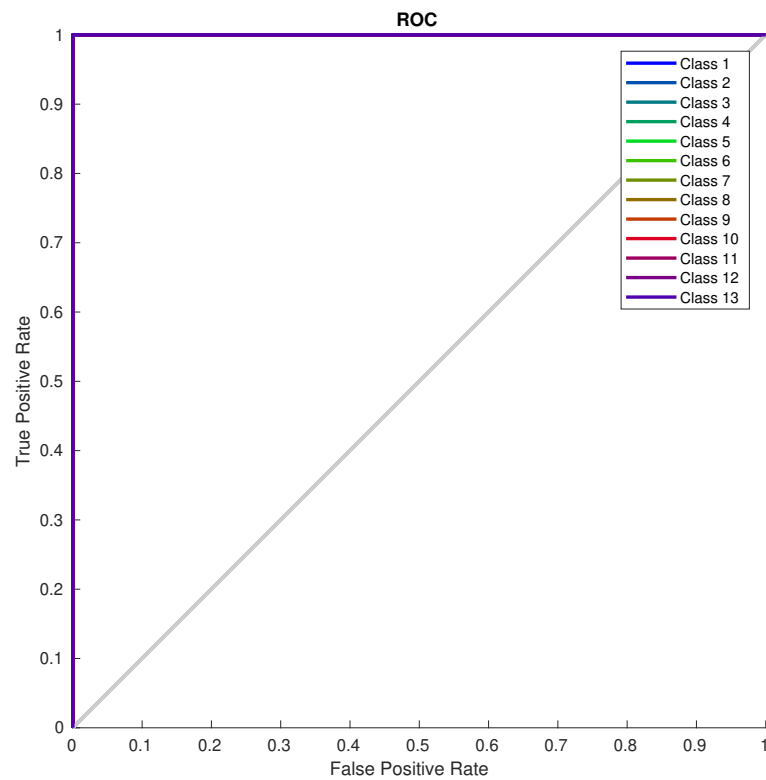


Figure 78 – ROC curve of the SVM_GPU Classifier - model with 35 parameters without the use of temperature and RBF kernel. AUC for class: class 1(R) - 1; class 2(A) - 1; class 3(B) - 1; class 4(C) - 1; class 5(F) - 1; class 6(I) - 1; class 7(L) - 1; class 8(D) - 1; class 9(E) - 1; class 10(G) - 1; class 11(H) - 1; class 12(J) - 1; class 13(K) - 1;