Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Asymmetric hidden Markov models and extensions applied to industry

PhD THESIS

Author

Carlos Esteban Puerto Santana
MSc Artificial Intelligence

2022

Thesis Committee

President: comite 1

External Member: comite 2

Member: comite 3

Member: comite 4

Secretary: comite 5

To my parents, Carlos and Carmen.

Acknowledgments

With this thesis, a stage of my academic life ends, and just as there were high and low points, there were attentive people who were there to support and congratulate me. In these lines I want to show my gratitude to those people and institutions that made these years something memorable.

To begin with, I want to thank my advisors, Pedro Larrañaga and Concha Bielza, who have taught me a lot and have guided me to be a researcher. I want to give thanks specially to Concha Bielza. Her comments, criticisms, and revisions taught me to doubt myself, my ideas, what I write, and to always seek for rigor and meaning in scientific research.

I must also acknowledge the support from Aingura IIoT and Inzu Group, companies that have allowed me to develop as a person and professional, especially Javier Díaz-Rozo who has helped me to land the ideas and methodologies that I have developed. Likewise, I would like to give thanks to the Barcelona Supercomputing Center, where Filippo Mantovani's group has been a very valuable and pertinent collaborator in completing this thesis. It is worth mentioning collaborators such as Guillem Ramirez or Carlos Bayona, who dedicated a lot of time and effort to put my ideas into practice. I also want to give thanks to Gustav Eje Henter from the KTH Royal Institute of Technology for his hospitality and for giving me new perspectives during my stay in Stockholm.

I thank those institutions and projects that partially funded this thesis: The Spanish Centre for the Development of Industrial Technology (CDTI) through the IDI-20180156 LearnIIoT project, the Spanish Ministry of Science and Innovation through the PID2019-109247GB-I00 and RTC2019-006871-7 DSTREAM projects, and from the project BAYES-CLIMA-NEURO, BBVA Foundation's Grant (2019). Also, by the H2020 IoTwins project (Distributed Digital Twins for industrial SMEs: a big-data platform) funded by the EU under the call ICT-11-2018- 2019, grant agreement number 857191.

Finally, I want to deeply thank my parents, Carlos and Carmen, who have always been there for me in the most difficult moments with their love and support. To my brother Cristian, who pushes me to be a better person every day. And to Daniela, a friend to my heart.

Agradecimientos

Con esta tesis, termina una etapa de mi vida académica, y así como hubo puntos altos y bajos, hubo también personas atentas que estuvieron allí para apoyarme y felicitarme. En estas lineas quiero escribir mis agradecimientos a aquellas personas e instituciones que hicieron de esta etapa algo memorable.

Para empezar, quiero darle las gracias a mis directores, Pedro Larrañaga y Concha Bielza, quienes me han enseñado mucho y me han guiado en mi formación como investigador. En especial, quiero agradecer a Concha Bielza, puesto que sus comentarios, criticas y revisiones, me enseñaron a dudar de mi mismo, de mis ideas, de lo que escribo y a buscar siempre el rigor y el sentido en la investigación científica.

Debo reconocer igualmente el apoyo por parte de Aingura IIoT e Inzu Group, empresas que me han permitido desarrollarme como persona y profesional, en especial a Javier Díaz-Rozo que me ha ayudado a aterrizar las ideas y metodologías que he desarrollado. Igualmente, agradezco al Barcelona Supercomputing Center, donde el grupo de Filippo Mantovani ha sido un colaborador muy valioso y pertinente para finalizar esta tesis. Cabe nombrar a colaboradores como Guillem Ramirez o Carlos Bayona, quienes dedicaron mucho tiempo y esfuerzo para llevar a la práctica mis ideas. También quiero agradecer a Gustav Eje Henter del KTH Royal Institute of Technology por su hospitalidad, y por darme nuevas perspectivas durante mi estancia en Estocolmo.

Agradezco a las instituciones y los proyectos que financiaron parcialmente esta tesis: el Centro para el Desarrollo Tecnológico Industrial (CDTI) con código IDI-20180156 proyecto LearnIIoT, el Ministerio de Ciencia e Innovación con código PID2019-109247GB-I00 y RTC2019-006871-7 DSTREAMS. También el proyecto BAYES-CLIMA-NEURO, BBVA Foundation's Grant (2019). Finalmente, el proyecto H2020 IoTwins (Distributed Digital Twins for industrial SMEs: a big-data platform) financiado por la Unión Europea, programa ICT-11-2018-2019, con número 857191.

Finalmente, quiero agradecer profundamente a mis padres, Carlos y Carmen, que siempre han estado allí para mí en los momentos más difíciles con su amor y apoyo. A mi hermano Cristian, que siempre me impulsa a ser una mejor persona cada día. Y a Daniela, una amiga para mi corazón.

Abstract

The Internet of things is a paradigm with the goal of creating customizable goods and services based on user experience. This paradigm has been applied in industrial environments generating what is called the industrial Internet of things. This new paradigm measures industrial assets continuously. The collected information is processed to extract data insights regarding the assets status or health. Depending on the asset status, maintenance policies can be planned to prevent the assets failure or degradation. To elaborate such insights, artificial intelligence models are usually applied to learn and predict industrial data patterns and behaviors. Nonetheless, in many cases, computational and reliability restrictions are present, and fast and explainable models are required to satisfy the industry needs.

Hidden Markov models (HMMs) are statistical models that are capable of learning data patterns and detect non-stationary behavior in data. When HMMs are compared to other models, HMMs are economic, explainable and reliable. They are economic because their learning and inference algorithms can run in a reasonable time without the need for graphic cards or other power-intensive computing devices. They are explainable since all the learned parameters are interpretable from a probabilistic and area-of-knowledge points of view. They are reliable, because, if a mistake is committed by the model, it is possible to detect and infer its causes from the models parameters and structure.

Due to the previous discussion, the motivation and results of this thesis aim to extend theoretically current HMMs, to make them more relevant, general and useful for industrial applications. For all the proposed models, the expectation-maximization algorithm was used for the learning phase. The first contribution appears in Chapter 4, where context-specific Bayesian networks were used to model the emission probabilities of continuous variables. That model is referred as AsLG-HMM since linear Gaussian Bayesian networks were used. The model was compared to a mixture of Gaussian HMM, where improvements in log-likelihoods by the proposed model were observed in both synthetic data and real data from ball-bearings. Nonetheless, such model was further developed in Chapter 5, where autoregressive values of the observable variables were considered in the context-specific Bayesian networks. This model is referred to AR-AsLG-HMM. In this case, the model was studied with further mathematical rigor. Also, a forward greedy algorithm was proposed to discover structures of Bayesian networks for the emission probabilities. The model was tested with synthetic and real data incoming from air quality and ball-bearing data. For this model, several types of HMMs were used for comparison. The learning times were also considered for evalua-

tion. The proposed model showed improvements in log-likelihood with fair learning times, and additional data insights were provided due to the learned Bayesian networks. As comment, AR-AsLG-HMM served as a cornerstone model for other contributions in this thesis. In Chapter 6, AR-AsLG-HMM was endowed with feature saliencies to enable the model to perform an embedded feature selection procedure. This implies that the model during its learning procedure determined the relevant features. This model is referred to FS-AsHMM. In this case, the model was compared to other HMMs with feature saliencies. Synthetic data and real data from ball-bearings and cameras with face- expressions data were used for validation. The model obtained better results regarding expression recognition and detection of non-relevant features.

The previous contributions were focused to offline analysis. Nevertheless, this thesis is focused on working in industrial environments, where data-streams are generated and the models are expected to adapt to changes in data. To address such issue, in Chapter 7 the AR-AsLG-HMM was adapted to be used in data-stream and perform continuous learning. Novel-concept detection techniques were used to determine when new unobserved patterns appeared. Based on the data-insights of the AR-AsLG-HMM from the data-stream, a health-index and a regression model were proposed to determine the health status and remaining useful life of ball-bearings. Two datasets were used to validate the proposed methodology: open access datasets with ball-bearings which are run to failure, and a ball-bearing testbed from a company promoting the thesis, Aingura IIoT. Additionally, in collaboration with the Barcelona Supercomputer Center, the methodology code was optimized to be embedded into edge devices and use it in real life applications. The methodology was compared to others in the state of the art. It obtained better results in terms of health estimation, and fair results regarding the remaining useful life prediction. Next, in Chapter 8 a feature saliency model for HMMs was adapted to determine relevant harmonics of ball-bearings data in online environments. However, this study was a preliminary work for what was done in Chapter 9, where local feature saliencies were applied on AR-AsLG-HMMs. This model is referred to LFS-AsHMM. This model was adapted to be used in data-streams with novel-concept detection techniques to keep track of the evolution of relevant features. This model updated the relevant features only when the data needed it. Synthetic and real open access data from ball-bearings was used for validation. The model was compared to other strategies and methodologies that perform feature selection in data-streams. However, these strategies did the feature selection whenever a new instance arrived and not when needed. Unfortunately, this model did not get to be implemented into edge devices during the writing of this thesis.

Finally, the proposed models assume linear Gaussian data and if such assumption fails, the models are no longer valid. To address such problem, in Chapter 10, the ideas used on AR-AsLG-HMM were imposed over HMMs with non-parametric emission probabilities, more precisely, kernel density estimations were used to approximate the emission probabilities, and the estimations depended on context-specific Bayesian networks. The proposed model is referred to KDE-AsHMM. The proposed model is validated using synthetic non-linear Gaussian data and open access real data from sound recognition problems and drill milling

processes. The model showed improvements in likelihood and sound recognition accuracy when compared to other HMMs. Nonetheless, the learning times and computational resources were high demanding. At the end of the thesis, in Chapter 11, the corresponding conclusions, final remarks and future research lines were proposed.

Resumen

El internet de las cosas es un paradigma con el objetivo de crear bienes y servicios perzonalizados basados en la experiencia del usuario. Este paradigma se ha aplicado en entornos industriales generando lo que se denomina el Internet de las cosas industriales. Este nuevo paradigma mide los activos industriales de forma continua. La información recopilada se procesa para extraer información sobre el estado o la salud de los activos. Según el estado de los activos, se pueden planificar políticas de mantenimiento para evitar fallas o degradación de los activos. Para elaborar tales conocimientos, modelos de inteligencia artificial se aplican para aprender y predecir patrones y comportamientos de datos industriales. No obstante, en muchos casos existen restricciones computacionales y de confiabilidad, y se requieren modelos rápidos y explicables para satisfacer las necesidades de la industria.

Los modelos ocultos de Markov (HMM) son modelos estadísticos que son capaces de aprender patrones de datos y detectar comportamientos no estacionarios en los datos. Cuando los HMMs se comparan con otros modelos, los HMMs son económicos, explicables y confiables. Son económicos porque sus algoritmos de aprendizaje e inferencia pueden ejecutarse en un tiempo razonable sin necesidad de tarjetas gráficas u otros dispositivos informáticos que consumen mucha energía. Son explicables ya que todos los parámetros aprendidos son interpretables desde un punto de vista probabilístico y de área de conocimiento. Son confiables, puesto que si el modelo comete un error, es posible detectar e inferir sus causas a partir de los parámetros y la estructura del modelo.

Debido a la discusión anterior, la motivación y los resultados de esta tesis tienen como objetivo extender los HMM teóricamente actuales, para hacerlos más relevantes, generales y útiles para aplicaciones industriales. Para todos los modelos propuestos se utilizó el algoritmo de maximización de expectativas para la fase de aprendizaje. La primera contribución aparece en el Capítulo 4, donde se utilizaron redes bayesianas específicas del contexto para modelar las probabilidades de emisión de variables continuas. Ese modelo se conoce como AsLG-HMM ya que se utilizaron redes lineales gaussianas bayesianas. El modelo se comparó con una mezcla de Gaussian HMM, donde se observaron mejoras en las probabilidades logarítmicas del modelo propuesto tanto en datos sintéticos como en datos reales de rodamientos de bolas. No obstante, dicho modelo se desarrolló más en el Capítulo 5, donde se consideraron los valores autorregresivos de las variables observables en las redes bayesianas específicas del contexto. Este modelo se denomina AR-AsLG-HMM. En este caso, el modelo fue estudiado con mayor rigor matemático. Además, se propuso un algoritmo voraz directo

para descubrir estructuras de redes bayesianas para las probabilidades de emisión. El modelo se probó con datos sintéticos y reales provenientes de la calidad del aire y datos de cojinetes de bolas. Para este modelo, se usaron varios tipos de HMM para comparar. Los tiempos de aprendizaje también fueron considerados para la evaluación. El modelo propuesto mostró mejoras en la probabilidad de registro con tiempos de aprendizaje justos, y se proporcionaron conocimientos de datos adicionales debido a las redes bayesianas aprendidas. Como comentario, AR-AsLG-HMM sirvió como modelo fundamental para otras contribuciones en esta tesis. En el Capítulo 6, se dotó a AR-AsLG-HMM con variables destacadas para permitir que el modelo realice un procedimiento de selección de variables incorporado. Esto implica que el modelo durante su procedimiento de aprendizaje determinó las variables relevantes. Este modelo se denomina FS-AsHMM. En este caso, el modelo se comparó con otros HMM con variables sobresalientes. Para la validación se utilizaron datos sintéticos y datos reales de rodamientos de bolas y cámaras con datos de expresiones faciales. El modelo obtuvo mejores resultados en cuanto al reconocimiento de expresiones y detección de variables no relevantes.

Las contribuciones anteriores estaban enfocadas al análisis fuera de línea. Sin embargo, esta tesis se centra en trabajar en entornos industriales, donde se generan flujos de datos y se espera que los modelos se adapten a los cambios en los datos. Para abordar este problema, en el Capítulo 7, el AR-AsLG-HMM se adaptó para usarse en flujo de datos y realizar un aprendizaje continuo. Se utilizaron técnicas de detección de conceptos novedosos para determinar cuándo aparecían nuevos patrones no observados. Con base en los conocimientos de datos del AR-AsLG-HMM del flujo de datos, se propusieron un índice de salud y un modelo de regresión para determinar el estado de salud y la vida útil restante de los rodamientos de bolas. Se utilizaron dos conjuntos de datos para validar la metodología propuesta: conjuntos de datos de acceso abierto con rodamientos de bolas que funcionan hasta el fallo y un banco de pruebas de rodamientos de bolas de la empresa promotora de la tesis, Aingura IIoT. Además, en colaboración con el Barcelona Supercomputing center, se optimizó el código de la metodología para integrarlo en edge devices y usarlo en aplicaciones de la vida real. La metodología fue comparada con otras en el estado del arte. Obtuvo mejores resultados en cuanto a la estimación de la salud, y resultados regulares en cuanto a la predicción de la vida útil remanente. A continuación, en el Capítulo 8, se adaptó un modelo de prominencia de variable para HMM para determinar los armónicos relevantes de los datos de rodamientos en entornos en línea. Sin embargo, este estudio fue un trabajo preliminar para lo que se hizo en el Capítulo 9, donde se aplicaron las prominencias de variables locales en AR-AsLG-HMM. Este modelo se denomina LFS-AsHMM. Este modelo se adaptó para usarse en flujos de datos con técnicas de detección de conceptos novedosos para realizar un seguimiento de la evolución de las variables relevantes. Este modelo actualizó las variables relevantes solo cuando los datos lo necesitaban. Para la validación se utilizaron datos de acceso abierto sintéticos y reales de rodamientos de bolas. El modelo se comparó con otras estrategias y metodologías que realizan la selección de variables en flujos de datos. Sin embargo, estas estrategias hacían la selección de funciones cada vez que llegaba una nueva instancia y no cuando era necesario. Desafortunadamente, este modelo no llegó a implementarse en dispositivos de *Edge* durante

la redacción de esta tesis.

Finalmente, los modelos propuestos asumen datos gaussianos lineales y si tal suposición falla, los modelos ya no son válidos. Para abordar tal problema, en el Capítulo 10, las ideas utilizadas en AR-AsLG-HMM se impusieron sobre los HMM con probabilidades de emisión no paramétricas, más precisamente, se utilizaron estimaciones de densidad kernel para aproximar las probabilidades de emisión, y las estimaciones dependían de redes bayesianas específicas del contexto. El modelo propuesto se refiere a KDE-AsHMM. El modelo propuesto se valida utilizando datos gaussianos no lineales sintéticos y datos reales de acceso abierto de problemas de reconocimiento de sonido y procesos de fresado de perforación. El modelo mostró mejoras en la probabilidad y la precisión del reconocimiento de sonido en comparación con otros HMM. No obstante, los tiempos de aprendizaje y los recursos computacionales fueron muy exigentes. Al final de la tesis, en el Capítulo 11, se propusieron las correspondientes conclusiones, comentarios finales y futuras líneas de investigación.

Contents

List of Figures

List of Tables

Acronyms and common symbols

Common acronyms:

- HMMs is hidden Markov models
- AR is auto regressive
- FSS is feature subset selection
- FS is feature saliency
- LGBN is linear Gaussian Bayesian network
- MVN is multiVariate normal distribution
- KDE is kernel density estimation
- RTF is run to failure
- CMAI is compute module Aingura insights
- BPFO is ball pass Frequency outer
- BPFI is ball pass Frequency inner
- FTF is fundamental train frequency
- BSF is ball spin frequency

Regarding random variables, densities and hyper parameters:

- $\boldsymbol{X}^t$ is the observable variables at time t
- Q^t is the hidden variable of a hidden Markov model at time t
- Z_m is the hidden feature relevancy variable of X_m
- Z_{im} is the hidden feature relevancy variable of X_m at $Q = i$
- $\mathbf{Pa}^t(X_m)$ is the set of non-AR parents of variable X_m in a graph at time t
- $\boldsymbol{u}_{im}^t$ is a vector sample of $\mathbf{Pa}^t(X_m)$ at time t and hidden state $Q^t = i$

- $\boldsymbol{d}_{im}^t$ is a vector sample of AR values at time t and hidden state $Q^t = i$
- $R(Q)$ range of the hidden variable Q
- N is the number of hidden states
- M is the number of features or variables
- p^* is the maximum permissible lag or AR values
- k_{im} is the number of parents of variable X_m at the hidden state $Q = i$
- p_{im} is the number of AR values for variable X_m at the hidden state $Q = i$
- f_{im} is the relevant probability distribution of X_m at $Q = i$
- g_m is the irrelevant probability distribution of X_m
- $g(i)$ is the hidden state labeling function evaluated at $Q = i$

A posteriori probabilities:

- $\alpha^t(i)$ is the forward variable evaluated at hidden state i at time t
- $\beta^t(i)$ is the backward variable evaluated at hidden state i at time t
- $\gamma^t(i)$ is the aposteriori probability of $Q^t = i$ at time t
- $\delta^t(i)$ is the most probable sequence of hidden states until $Q^t = i$
- $\psi_m^t(i)$ is the aposteriori probability of $Q^t = i$ and $Z_m^t = 1$
- $\phi_m^t(i)$ is the aposteriori probability of $Q^t = i$ and $Z_m^t = 0$

Common parameters found along the thesis:

- $\mathcal{B}$ is the graph of a Bayesian network
- $\boldsymbol{\theta}$ denotes a set of parameters
- $\boldsymbol{\lambda}$ is the set of parameters of a hidden Markov model
- $\mathcal{B}_{q^t}$ or $\mathcal{B}_i$ is the context specific graph of the hidden state $Q^t = i$
- $\mathbf{A}$ is the transition matrix of a hidden Markov model
- $\mathbf{B}$ is the emission probabilities of a hidden Markov model
- $\boldsymbol{\pi}$ is the initial distribution of hidden states
- ρ_m is the feature saliency of feature X_m
- ρ_{im} is the feature saliency of feature X_m at $Q^t = i$

- $\boldsymbol{\beta}_{im}$ are the weights of $\boldsymbol{u}_{im}^t$ that explain X^t at time t at $Q^t = i$
- $\boldsymbol{\eta}_{im}$ are the weights of $\boldsymbol{d}_{im}^t$ that explain X^t at time t at $Q^t = i$
- $\varphi_{im}^t = \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ is the mean of X_m at time t at $Q^t = i$

Some relevant miscellaneous terms:

- $\mathcal{Q}$ is the auxiliary function of the EM algorithm
- (s) is the index of the EM algorithm iteration
- s is a Page function
- $\chi_A(x)$ is the indicator function; it is one if $x \in A$, zero otherwise
- χ_c is the indicator function, which is 1 if c is true and 0 otherwise

Chapter 1

Introduction

1.1 Motivation

Nowadays the use of artificial intelligence is being more and more extensive in different daily life areas. The increasing capacity in data-storage, data processing speed and Internet coverage, have enabled companies, governments and individuals, to obtain, storage and process huge amounts of data. With the introduction of the Internet of things paradigm, goods and services have mechanisms to extract data from their usage and condition. The extracted data is later analyzed in order to generate better and more customized products for future or current clients. In particular, an interesting application of this paradigm is the industrial Internet of things with the 4.0 industrial revolution (4IR). The 4IR proposes to sensor industrial assets and production/delivery lines in order to generate actionable insights (Larrañaga et al. [2018]). However, to obtain such actionable insights, it is required to capture, store, process and analyze data, and this routine may be done in the industrial asset itself or in an external server. In the first case, we talk about edge computing: it requires the incorporation of edge devices as small computers embedded or close to the asset itself; this computer can be used when the data processing is not complex and does not require huge amounts of data. In the second case, we talk about cloud computing: this paradigm is an option when the required data storage is important and the algorithms to analyze and process the data are highly time and memory consuming. In both cases, as an industrial asset or delivery/production line may be established in a position far from the control center location, the use of new Internet technologies such as 4G or 5G can be useful for the data or actionable insights transmission and communication.

It is desirable for an industrial asset or production/delivery line to prevent anomalies due to failures or degradation (Tseng [1996]). Within industrial environments, ball bearings are components which can be found in almost all rotating machinery, and they are commonly associated with premature failures. An unexpected failure of these components can be critical enough to stop a production system completely (Mobley [2002]). In spite that these are designed to not fail under certain given mechanical and thermal constrains, these can fail or reduce their remaining useful life (RUL) due to changes in their operating conditions (Sutrisno

et al. [2012]). Given this scenario, artificial intelligence via machine learning techniques can be applied to detect and avoid undesired situations in industrial assets or production/delivery lines due to a ball bearing failure. In this case, an intelligent system should be able to estimate the RUL without expensive and extensive training and inference procedures (Skariah et al. [2020]).

In many cases, the failing mode of an industrial asset or mechanical component as a ball bearing change abruptly even under the same industrial constraints. A predictive machine learning algorithms have to deal with highly unbalanced data, as there are not enough failures to train a standard predictive model. Therefore, a machine learning-based ball bearing health assessment system has the following specific challenge to overcome: to detect and adapt to low degradation rates, unbalance data training cases and to be applicable and robust for machines with extensive lifespans. A possible solution to these issues is to use adaptive models which update themselves when previously unknown patterns are being observed in data streams. The identification of such trends is pertinent for a model or AI performance in an online environment, where learned trends may become obsolete (Markou and Singh [2003]). Then, an intelligent system dedicated to estimate and predict the ball bearing RUL, should be able to self-adapt, detect new patterns in data and use the newest trends to re-estimate the RUL prediction. Furthermore, the model used must run its learning, prediction, and novel pattern detection algorithms as soon as data arrives to be feasible for industrial applications. Under such circumstances, hidden Marjov models (HMMs) are a candidate machine learning model to be used. HMMs are probabilistic graphical models where time dynamics are taken into consideration and are capable of model non stationary data (Rabiner [1990]). Due to its simplicity, they do not require large amount of time and memory resources for their learning and inference tasks, which is desirable when working in edge computing environments. Additionally, as it is a probabilistic graphical model, its parameters and structure can be interpreted and failures and data insights can be more easily explained.

1.2 Objective and methodology

From the previous discussion, the following question arises:

Is it possible to define an adaptive model or methodology based on HMMs capable to work online to estimate ball bearings health and RUL?

The previous question comes with several functional requirements that arise when the methodology or model is expected to work with data streams in edge devices. The functional requirements are:

A. The model must be fast in the learning and inference phases

B. The model must be able to detect novel concepts in data streams and adapt to them

C. The model must be able to estimate the industrial asset (ball bearing) health and remaining useful life (RUL)

D. The model must work in unsupervised environments, since not enough failure labeled data is usually available

E. The model must determine the most relevant features in an online manner

F. The model must be interpretable

This thesis will focus on exploring the answer to the previous question taking into account the functional requirements. To do so, various extensions of HMMs are proposed in order to obtain and extract additional data insights. These new HMMs are defined theoretically and their training and inference algorithms are explained with mathematical detail and rigor. The computational complexity of the algorithms is taken into consideration, especially for the training phase, since it was noted from previous works e.g., Diaz-Rozo et al. [2020], that this phase was a bottleneck for online applications. Then, datasets coming from mechanical setups with ball bearings are used to validate the models. To show the capabilities of the proposed models and methodologies, they are compared to others from the state of the art. In this sense, it is proven that the advantages and developments that this thesis is proposing are not irrelevant. However, for the sake of generality, the models are also tested with other kind of data such as air pollution data, grammatical face expression and sound records data sets. In this manner, it is also demonstrated that the proposed models are not ad hoc for ball bearing data.

1.3 Thesis organization

The thesis is divided into three parts: foundations , contributions and conclusions. The foundations (Part I) have two chapters:

- Chapter 2 contains the theoretic tools to understand this thesis. Basics of Bayesian networks, HMMs, kernel density estimation and novelty detection. Special attention is given to asymmetric HMMs, since from these models, all the contributions are developed.

- Chapter 3 reviews the state of the art related with HMMs, their asymmetric version, methods for machine-tool prognosis, offline and online feature subset selection and kernel density estimation on HMMs.

In the contributions (Part II) there are seven chapters.

- Chapter 4 introduces the first model extension of asymmetric HMMs; the model enables asymmetric HMMs to deal with Gaussian variables.

- Chapter 5 extends furthermore the capabilities of asymmetric HMMs, allowing observable variables to depend on previous observations.

- Chapter 6 enables asymmetric HMMs to determine the most relevant features in an embedded manner, i.e., the features are selected during the learning process.

- Chapter 7 is focused on applying the asymmetric HMMs to online prognosis of ball bearings in data streams inside edge devices. This chapter can be pointed out as the core of this thesis, since the main research question and several of the functional requirements are explored.

- Chapter 8 proposes a first approach to perform feature subset selection in online environments using HMMs.

- Chapter 9 extends the ideas proposed in Chapters 6 and 8, where the proposed model is capable of determining the most relevant features for each data instance in a data-stream environment.

- Chapter 10 serves as a final extension of the ideas proposed in Chapter 5, which provides a possible answer to the question: *How can the proposed models be extended to deal with non linear Gaussian data?* The solution is based on kernel density estimations.

Later, in the conclusions (Part III), a single chapter is found. Chapter 11 provides an answer to the main question of the thesis considering its restrictions. Strengths and drawbacks from each of the proposed models and methodologies are reviewed to determine the level of satisfaction with the functional requirement constraints. Also, open issues and possible future research lines are discussed.

Finally, five appendices are found at the end of the thesis with proofs of theorems, sensitivity analyses of the prognosis model and parameters for synthetic data.

1.4 Notation

In this thesis, the following conventions are made: boldface symbols are used to denote vectors: X is a scalar, whereas $\boldsymbol{X}$ is a vector or matrix. Capital letters are used to denote random variables, and small letters are used to refer to values of the random variables: X is a random variable and x is a sample from such variable. The range of a random variable X is denoted as $R(X)$. For the sake of space $\sum_{x \in R(X)} f(x)$ is written as $\sum_{R(X)} f(x)$. The symbol $\top$ will be used to denote matrix transpose, therefore $(X^1, ..., X^T)^\top$ is a column vector of size T. The notation $\boldsymbol{X}^{1:T}$ refers to the row vector $(X^1, ..., X^T)$ of size T. If $\boldsymbol{X}^t = (X_1^t, ..., X_M^t)$ is a row vector of size M, then $\boldsymbol{X}^{1:T} = (\boldsymbol{X}^1, ..., \boldsymbol{X}^T)^\top$ is a matrix of size $T \times M$. If $\boldsymbol{x}$ is a row vector of size T and $\boldsymbol{y}$ is another row vector of size L, then, the concatenation of vectors is denoted as $[\boldsymbol{x}|\boldsymbol{y}]$, this new row vector has size $T+L$. The notation $\text{Set}(\boldsymbol{X}^{1:T}) = \{X^1, ..., X^T\}$ indicates that the vector $\boldsymbol{X}^{1:T}$ is being used as a set. If A is a set and $A \subset B$, then A^c refers to the complement of A in B or $A^c = B \backslash A$. Finally, if $\boldsymbol{X}^{1:T}$ is a vector of size T, then $\text{Matrix}(\boldsymbol{X}^{1:T})$ is a matrix of zeros of size $T \times T$, but the diagonal corresponds to $\boldsymbol{X}^{1:T}$.

Part I

FOUNDATIONS

Chapter 2

Theoretical background

In real world problems, it is desirable to measure the probability of future or current events given a set of previously observed ones. A common assumption to compute such probabilities is to assume that each event is independent of the others, and the observed variables corresponding to the event are also independent, more formally: if $\boldsymbol{X}^{0:T} = (\boldsymbol{X}^0, ..., \boldsymbol{X}^T)$ is a list of events with $\boldsymbol{X}^t = (X_1^t, \cdots, X_M^t)$, then, the independence assumption factorizes the joint distribution as follows:

$$P(\boldsymbol{X}^{0:T}) = \prod_{t=0}^{T} \prod_{m=1}^{M} P(X_m^t), \tag{2.1}$$

in spite that this assumption eases the computation and estimation of probabilities, it is known in economics, physics, engineering, biology and so on, that time dependencies and probabilistic relationships among variables is latent and should not be omitted. A way to overcome this issue is using Bayesian networks and their extensions, which serves as an approximation to considerate probabilistic dependencies. This chapter will provide the fundamental tools to read and understand the contributions of this thesis. The tools are focused on modeling probabilistic time series, kernel density estimation, and detect novel trends in data using dynamical Bayesian networks. For a deeper scope in Bayesian networks and dynamic Bayesian networks, the reader is referred to Koller and Friedman [2009] and Murphy [2002]. Additionally, some foundations signal processing and time series are provided in order to work with data from industrial assets.

2.1 Bayesian networks

2.1.1 Fundamentals

This section begins with a important concept related to probabilistic dependencies:

Definition 2.1. Let X, Y and Z be random variables. X is **conditionally independent**

of Y given Z if:

$$P(X|Y,Z) = P(X|Z) \tag{2.2}$$

This concept can be used to factorize the joint probability function considering probabilistic relationships among variables. The joint probabilistic function without any simplification can be written as:

$$\begin{aligned} P(\boldsymbol{X}^t) &= P(X_1^t, ..., X_M^t) \\ &= P(X_1^t | X_2^t, \cdots, X_M^t) P(X_2^t, ..., X_M^t) \\ &= \prod_{m=1}^{M} P(X_m^t | \boldsymbol{X}_{m+1:M}^t), \end{aligned} \tag{2.3}$$

Observe that this factorization relies in the index ordering and therefore can be different. However, this representation is not desired. As instance, in the binary case, i.e., each variable is Bernoulli, each factor $P(X_m^t|\boldsymbol{X}_{m+1:M}^t)$ requires to store and estimate $2^{M-m+1}-1$ probabilities. It is pertinent to reduce the number of parameters and dependencies. Taking into consideration the conditionally independent property, each factor $P(X_m^t|\boldsymbol{X}_{m+1:M}^t)$ can reduce its dependencies. Let $\mathbf{Pa}^t(X_m^t) \subseteq \text{Set}(\boldsymbol{X}_{m+1:M}^t)$ be the set of parents of X_m^t or the subset of variables with the following property:

$$\begin{aligned} P(X_m^t|\boldsymbol{X}_{m+1:M}^t) &= P\left(X_m^t|\mathbf{Pa}^t(X_m^t), (\mathbf{Pa}^t(X_m^t))^{\mathbf{c}}\right) \\ &= P(X_m^t|\mathbf{Pa}^t(X_m^t)). \end{aligned} \tag{2.4}$$

In this sense, $(\mathbf{Pa}^t(X_m^t))^{\mathbf{c}}$ is the set of variables conditionally independent of X_m^t given $\mathbf{Pa}^t(X_m^t)$. The set $\mathbf{Pa}^t(X_m^t)$ contains the relevant dependencies of X_m^t for the given index ordering. The factorization of the joint probability function would be:

$$P(\boldsymbol{X}^t) = \prod_{m=1}^{M} P(X_m^t|\mathbf{Pa}^t(X_m^t)) \tag{2.5}$$

An example is provided to illustrate the advantages of using conditional independencies. The factorization in Eq. (2.3) can be pictured as a directed graph in the following manner: each node of the graph is a variable X_m^t, if X_m^t depends on X_n^t, and arc form X_n^t to X_m^t is drawn. For $m = 4$, the factorization in Eq. (2.3) is pictured. Observe that no loop is generated in the graph, and therefore, the graph is actually a directed acyclic graph (DAG). Let assume that X_1^t is independent of X_4^t given X_3^t and X_2^t, X_2^t is independent of X_3^t given X_4^t, and X_3^t is independent of X_4^t. With these considerations, assuming additionally that the variables are Bernoulli, the number of parameters is reduced from $2 + 2^2 + 2^3 + 2^4 - 4 = 26$ to $2 + 2 + 2^2 + 2^3 - 4 = 12$. The reduced graph considering the mentioned conditional independencies is drawn in Fig. 2.1.

Now, a useful definition (D-separation) is introduced. It will be useful to characterize conditional independencies in arbitrary graphs:

Definition 2.2. A set of nodes $\mathcal{Z}$ **D-separates**two sets of nodes $\mathcal{X}$ and $\mathcal{Y}$ in a DAG if, for

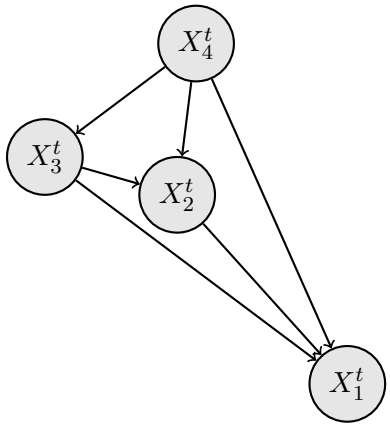


Figure 2.1: Inferred graph from factorization from Eq. (2.3)

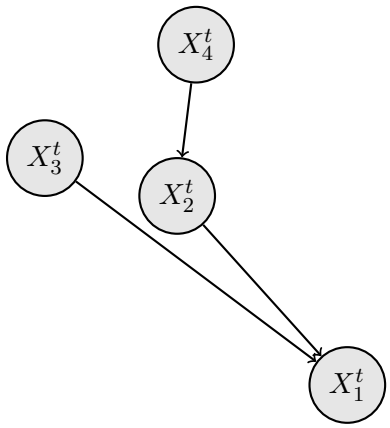


Figure 2.2: Inferred graph after introducing conditional independencies

every pair of nodes $x \in X$ and $y \in Y$ connected by an undirected path, there is a node c in such path that fulfills one of the followings:

1. c is a converging node in the directed graph with c and its descendants do not belong to $\mathcal{Z}$.

2. c is not a converging node and $c \in \mathcal{Z}$.

In Koller and Friedman [2009] it is proven that if two nodes x and y are D-separated given a set $\mathcal{Z}$ in a DAG, they are conditionally independent. Usually, a DAG model for a set of variables is proposed and the D-separation property is used to determine for each variable its conditional independencies. Now, with these definitions and the previous concepts being cleared, a fundamental definition for this thesis is introduced:

Definition 2.3. Given a set of random variables $\text{Set}(\boldsymbol{X}^t)$, a **Bayesian network** is a tuple $(P, \mathcal{B})$, where $\mathcal{B}$ is a DAG with the following properties.

1. The nodes of $\mathcal{B}$ are the elements of $\text{Set}(\boldsymbol{X}^t)$.

2. Each node is conditionally independent of its non-descendants given its parents.

3. Each node has a local probability $P\left(X_m^t|\mathbf{Pa}^t(X_m^t)\right)$ such that the joint probability is factorized as $P\left(\boldsymbol{X}^t\right) = \prod_{m=1}^{M} P\left(X_m^t|\mathbf{Pa}^t(X_m^t)\right)$.

4. D-separated variables in the graph are independent.

Bayesian networks are a useful tool to provide probabilistic models to describe and predict random variables and stochastic processes. Due to its structure, it can provide insights of the nature of the problem and the probabilistic relationships among variables. It also reduces the number of parameters, which reduces the risk of data overfitting.

2.1.2 Kahn topological ordering

When working with Bayesian networks, it is required for some inference task to visit every node and have information from all the parent nodes. To ensure that, when passing over a node, the information of its parents is already stored, a topological ordering can be imposed.

A topological ordering is an ordered list which goes from the nodes with less dependencies to the ones with the most dependencies. The algorithm described by Kahn [1962] provides a topological ordering for a directed graph $\mathcal{B}$ that also provides a condition to ensure that $\mathcal{B}$ has no cycles. In the case of Bayesian networks, it is pertinent to ensure that a given graph is a DAG. The main idea of the algorithm is to add a variable to the ordering list only if all its parents have been already added. A pseudo code of such strategy is presented in Algorithm 2.1.

Algorithm 2.1 Topological sorting

Input: A directed graph $\mathcal{B}$.
Return: A list L with a topological ordering.

```
Set L a zero dimensional vector
Set S a set with the nodes of B with not incoming arrows.
while S ≠ ∅ do:
    Choose s ∈ S.
    Set L := [s, L] (concatenation).
    for q which s ∈ Pa(q) do:
        Set Pa(q) := Pa(q)\{s}.
        if Pa(q) = ∅ then:
            Set S := S ∪ {q}.
if {q : Pa(q) ≠ ∅} ≠ ∅ then:
    return L and B has a cycle and is not a DAG
else:
    return L and B is a DAG
```

2.1.3 Linear Gaussian Bayesian networks

Linear Gaussian Bayesian networks (LGBNs) are Bayesian networks designed to work with continuous variables. However, it is assumed that the variables are related in a linear manner

and the error follows a Gaussian distribution. Therefore, the marginals, joint and conditional distributions are Gaussian.

Definition 2.4. Let $\boldsymbol{X}^t = (X_1^t, ..., X_M^t)$ be a continuous random vector. A **linear Gaussian Bayesian network** over $\boldsymbol{X}^t$ is a a Bayesian network $(\boldsymbol{\theta}, \mathcal{B})$, where $\mathcal{B}$ is a directed acyclic graph (DAG). The parents of X_m^t are given by $\mathcal{B}$ and are represented by an ordered vector of length k_m denoted as $\mathbf{Pa}(X_m^t) = (U_{m,1}^t, ..., U_{m,k_m}^t)$, $m = 1, ..., M$, with $U_{m,l}^t \in \text{Set}(\boldsymbol{X}^t)$, $l = 1, ..., k_m$. $\boldsymbol{\theta}$ is formed by the local distributions of each X_m^t conditioned on $\mathbf{Pa}(X_m^t)$. The joint density probabilistic function satisfies:

$$\begin{aligned} f_{\boldsymbol{X}^t}(\boldsymbol{X}^t) &= \prod_{m=1}^{M} f_{\boldsymbol{X}_m^t}(X_m^t | \mathbf{Pa}^t(X_m^t)) \\ &= \prod_{m=1}^{M} \mathcal{N}(X_m^t | \beta_{m,0} + \beta_{m,1} U_{m,1}^t + \cdots + \beta_{m,k_m} U_{m,k_m}^t, \sigma_m^2) \\ &= \prod_{m=1}^{M} \mathcal{N}(X_m^t | \boldsymbol{U}_m^t \boldsymbol{\beta}_m, \sigma_m^2) \end{aligned} \tag{2.6}$$

In the previous equation $\boldsymbol{\beta}_m = (\beta_{m,0}, \beta_{m,1}, ..., \beta_{m,k_m})^\top$ and $\boldsymbol{U}_m^t = (1, U_{m,1}^t, ..., U_{m,k_m}^t)$.

Sometimes, it is required to compute some information measures such as divergence, entropy and so on. Many of such measures have closed formula for Gaussian distributions but not for LGBN. The next theorem with its proof in Koller and Friedman [2009], gives us a tool to transform LGBN in multivariate Gaussian distributions (MVG).

Theorem 2.1. Let X_m^t be a node of a LGBN with parents $\boldsymbol{U}_m^t$, weights $\boldsymbol{\beta}_m$ and variance σ_m^2. Assuming that $\boldsymbol{U}_m^t$ are jointly Gaussian with normal distribution $\mathcal{N}(\boldsymbol{\mu}_U, \boldsymbol{\Sigma}_U)$. Then, the following holds:

1. The distribution of X_m^t is a normal distribution $P(X_m^t) = \mathcal{N}(\mu_{X_m^t}, \sigma_{X_m^t}^2)$:

$$\begin{aligned} \mu_{X_m^t} &= \boldsymbol{\mu}_U \boldsymbol{\beta}_m \\ \sigma_{X_m^t}^2 &= \sigma_m^2 + \boldsymbol{\beta}_m^\top \boldsymbol{\Sigma}_U \boldsymbol{\beta}_m \end{aligned} \tag{2.7}$$

2. The joint distribution over $[X_m^t | \boldsymbol{U}_m^t]$ is a normal distribution where its covariance matrix is computed as:

$$\text{Cov}[X_m^t, U_{m,l}^t] = \sum_{k=1}^{k_m} \beta_{m,k} [\boldsymbol{\Sigma}_U]_{l,k} \quad l = 1, ..., k_m \tag{2.8}$$

As corollary, if the topological sorting algorithm given in Section 2.1.2 is used an a order of the graph is used to apply Eq. (2.7) and Eq. (2.8). Then the mean vector and covariance matrix of all the nodes of a LGBN are computed and a MVG is obtained.

2.1.3.1 Learning the parameters of a LGBN

To learn the parameters of a LGBN, for each node X_m^t, the set of parameters $\boldsymbol{\beta}_m$ and σ_m^2 need to be estimated. Estimation techniques such as the maximum likelihood estimation (MLE) can be applied for this case. Assume that a sample $\boldsymbol{x}^{0:T}$ has been measured. The MLE estimator can be found solving:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ln\left(P(\boldsymbol{x}^{0:T}|\boldsymbol{\theta})\right) \tag{2.9}$$

In our case, the set of parameters $\boldsymbol{\theta} = \{\boldsymbol{\beta}_m, \sigma_m^2\}_{m=1}^M$ such that maximizes the log-likelihood must be estimated:

$$\begin{aligned} \ln\left(f_{\boldsymbol{X}^{0:T}}(\boldsymbol{x}^{0:T}|\boldsymbol{\theta})\right) &= \ln\left(\prod_{t=0}^{T} f_{\boldsymbol{X}^t}(\boldsymbol{x}^t|\boldsymbol{\theta})\right) = \ln\left(\prod_{t=0}^{T}\prod_{m=1}^{M} \mathcal{N}(x_m^t|\boldsymbol{u}_m^t\boldsymbol{\beta}_m, \sigma_m^2)\right) \\ &= \sum_{m=1}^{M}\sum_{t=0}^{T} \ln\left(\mathcal{N}(x_m^t|\boldsymbol{u}_m^t\boldsymbol{\beta}_m, \sigma_m^2)\right) \end{aligned} \tag{2.10}$$

Deriving the previous equation with respect to the parameters $\boldsymbol{\beta}_m$ and σ_m^2, equating to zero and solving with respect to the parameters, the following estimating formulas are obtained:

$$\begin{aligned} \hat{\boldsymbol{\beta}}_m &= \left((\boldsymbol{u}_m^{0:T})^\top \boldsymbol{u}_m^{0:T}\right)^{-1} (\boldsymbol{u}_m^{0:T})^\top \boldsymbol{x}_m^{0:T} \\ \hat{\sigma}_m^2 &= \frac{1}{T+1}\sum_{t=0}^{T}(x_m^t - \boldsymbol{u}_m^t\hat{\boldsymbol{\beta}}_m)^2 \end{aligned} \tag{2.11}$$

The estimation formulas are the same as in the case of ordinary least square (OLS) problems. Observe that it was assumed that each sample is independent of the others and no temporal probabilistic dependencies are taken. Once a set of parameters has been learned, the likelihood computation of a new sample follows from using Eq. (2.6).

2.1.4 Kernel density estimation

2.1.4.1 Motivation

In many scenarios, due to the central limit theorem, the Gaussian distribution is well fitted to explain and model data distribution from continuous stochastic variables. Nevertheless, in other cases, the observed data distribution do not correspond to a Gaussian one, as instance, the waiting times for an event to happen are better modeled using an exponential distribution, the sum of waiting times behaves as a gamma distribution, and so on. Nonetheless, in many cases, it is not known the nature of the data distribution and it must be estimated in some non parametrical manner.

The first approximation to estimate an unknown distribution is using histograms. But, this approximation has many issues. The number, width and position of the bin edges must be deduced and the final estimation is not continuous nor smooth Silverman [1986]. A more

sophisticated and general approximation was introduced in Rosenblatt [1956] and Parzen [1962], where the data distribution approximation was assumed to be a sum of delta Dirac equations, and the estimation of the probability density function (pdf) is the derivative of it. As result of this hypothesis, the density was estimated using a mixture of uniform distributions, weighted by a bandwidth h (similar to the bin width in an histogram). Nonetheless, this mixture was generalized in the following manner: let $K : \mathbb{R} \to \mathbb{R}$ be a symmetric pdf, also known as the kernel, $h > 0$ a bandwidth, $D = (y^0, ..., y^T)$ a training dataset i.i.d from a random variable Y, a **kernel density estimation** (KDE) of the pdf $f_Y(y)$ is defined as:

$$\hat{f}_Y(y|D,h) = \sum_{t=0}^{T} \frac{1}{nh} K\left(\frac{y - y^t}{h}\right). \tag{2.12}$$

2.1.4.2 Kernel and bandwidth selection

There are many issues to solve when working with kernels, such as selecting the kernel function K and the bandwidth h. Usually, K is the normal distribution, i.e., $K(x) = (2\pi)^{-1}e^{-\frac{x^2}{2}}$, this kernel is selected due to its well known properties as smoothness and unbounded support. However, in Parzen [1962], the selection for K is not crucial in terms of asymptotic behavior. It was proven that the approximation $\hat{f}_Y$ is asymptotically unbiased, if $h \to 0$ when $T \to \infty$ and K has the following properties:

1. $\sup_{|y|<\infty} |K(y)| < \infty$ (bounded)

2. $\int_{-\infty}^{\infty} |K(y)|dy < \infty$ (finite absolute integral)

3. $\lim_{y\to\infty} |yK(y)| = 0$ (K annihilates y in the limit)

Additionally, it was proven the following two limit properties:

1. $\lim_{n\to\infty} \mathrm{Var}[\hat{f}_Y(y|D,h)] = (nh)^{-1} f_Y(y) \int_{-\infty}^{\infty} K^2(x)dx$ (limit of the pointwise variance)

2. $\lim_{n\to\infty} E\left[\left(\hat{f}_Y(y|D,h) - f_Y(y)\right)^2\right] = 0$ (quadratic mean consistency)

Nonetheless, K can be constructed to ensure the minimization of the error of the estimation, given that a proper h has been chosen. In Rosenblatt [1956], h is selected as solution of minimizing the mean integral squared error (MISE):

$$\begin{aligned} \mathrm{MISE}(h) &= E\left[\int_{-\infty}^{\infty} \left(\hat{f}_Y(y|D,h) - f_Y(y)\right)^2 dy\right] \\ &= \int_{-\infty}^{\infty} \left(E[\hat{f}_Y(y|D,h)] - f_Y(y)\right)^2 dy + \int_{-\infty}^{\infty} \mathrm{Var}[\hat{f}_Y(y|D,h)]dy \end{aligned} \tag{2.13}$$

Assuming that the first statistical moment of K is zero, its second moment is finite, not null and $f_Y(y)$ is smooth, then, the following approximations can be made (using Taylor series):

$$\begin{aligned} E[\hat{f}_Y(y|D,h)] - f_Y(y) &\approx \frac{1}{2}h^2 f_Y''(y)dy \int_{-\infty}^{\infty} x^2 K(x)dx, \\ \mathrm{Var}[\hat{f}_Y(y|D,h)] &\approx (nh)^{-1} f_Y(y) \int_{-\infty}^{\infty} K^2(x)dx, \end{aligned} \tag{2.14}$$

and therefore, the approximate MISE (AMISE) can be defined and used as objective function for determining h:

$$\mathrm{AMISE}(h) = \frac{1}{4}h^4 \int_{-\infty}^{\infty} f_Y''(y)^2 dy \left(\int_{-\infty}^{\infty} x^2 K(x)dx \right)^2 + (nh)^{-1} \int_{-\infty}^{\infty} K^2(x)dx \tag{2.15}$$

For this function, the minimization is easily done, but the solution h^* depends on $f_Y''(y)$ which is unknown:

$$h^* = \left(\frac{\int_{-\infty}^{\infty} K^2(x)dx}{n \int_{-\infty}^{\infty} f_Y''(y)^2 dy (\int_{-\infty}^{\infty} x^2 K(x)dx)^2} \right)^{1/5} \tag{2.16}$$

In Jones et al. [1996] many strategies to approximate $f_Y''(y)$ or to minimize the MISE or AMISE function are reviewed and compared. If h^* is introduced in Eq. (2.15), the optimal approximate mean integrated square error is:

$$\mathrm{AMISE}(h^*) = \frac{5}{4n^{4/5}} C(K) \left(\int_{-\infty}^{\infty} f''(x)dx \right)^{1/5}. \tag{2.17}$$

Where $C(K) = \left(\int_{-\infty}^{\infty} x^2 K(x)dx\right)^{2/5} \left(\int_{-\infty}^{\infty} K(x)^2 dx\right)^{4/5}$. It can be noted that minimizing $C(K)$, also minimizes the AMISE value. Minimizing $C(K)$ with constrains $\int K(x) = \int x^2 K(x) = 1$ leads to the the Epanechnikov kernel Epanechnikov [1969]:

$$K_e(x) = \begin{cases} \frac{3}{4\sqrt{5}} \left(1 - \frac{x^2}{5}\right), & |x| \leq \sqrt{5} \\ 0, & |x| > \sqrt{5} \end{cases} \tag{2.18}$$

It can be noted that the optimal kernel $K_e(x)$ is not smooth and has compact support. For some applications, these properties can be troublesome. It has been proven that other kernels (e.g. Gaussian, rectangular, triangular, e.t.c.) are almost as efficient as $K_e(x)$ in terms of $C(K)$ Silverman [1986]. Therefore, as long as K fulfills the conditions to guarantee a consistent estimation of the density, K can be chosen as desired depending on the application.

All the previous discussion about estimations and properties can be easily generalized to the case of multidimensional data as stated in Silverman [1986], where the KDE can be defined as:

$$\hat{f}_{\boldsymbol{Y}}(\boldsymbol{y}|D,h) = \frac{1}{nh^M} \sum_{t=0}^{T} K\left(\frac{\boldsymbol{y} - \boldsymbol{y}^t}{h}\right) \tag{2.19}$$

2.1.4.3 Bayesian networks based on KDE

In Silverman [1986], it is exposed a table with the minimum number of instances required to obtain a 1% in the KDE relative square error estimation. This error was measured for a single point of a MVG distribution for different dimensions. For example, 223 instances are required in four dimensions, 43700 instances in eight dimensions and 842000 in ten dimensions. It is observed, that the amount of instances required to obtain an error of 1% (for a single point estimation), increases rapidly with the dimension. Which implies that KDE is not suitable to be used in high dimensions. However, it is reasonable to think that a Bayesian network representation can be used to alleviate this issue and make KDE estimations scalable to high dimensions. Recall that in a Bayesian network, the joint probability distribution is factorized using conditional independencies, and therefore, for each variable X_m^t, its conditional probability $P(x_m^t|\mathbf{Pa}(x_m^t)))$ must be computed. In the case of KDE, the conditional dependencies can be computed using directly the definition of conditional pdf Pérez et al. [2009]. Or more formally:

$$\hat{f}_{X_m}(X_m^t|\mathbf{Pa}(X_m^t)) = \frac{\hat{f}_{X_m,\mathbf{Pa}(X_m)}(X_m^t, \mathbf{Pa}(X_m^t))}{\hat{f}_{\mathbf{Pa}(X_m)}(\mathbf{Pa}(X_m^t))} \tag{2.20}$$

In this case, for each factor, two densities must be estimated using KDEs: the joint probability of a variable and its parents and the marginal of its parents. Therefore, the estimation of the joint pdf of $\boldsymbol{X}^t$ would be:

$$\hat{f}_{\boldsymbol{X}}(\boldsymbol{X}^t) = \prod_{m=1}^{M} \hat{f}_{X_m}(X_m^t|\mathbf{Pa}(X_m^t)) \tag{2.21}$$

From Eq. (2.20), it is deduced that a KDE estimation in a Bayesian network is reasonable when each variable has a little amount of parents. Otherwise, the curse of dimension would make the estimation highly unbiased and expensive to compute. In Pérez et al. [2009], various structure learning algorithm were used to learn the Bayesian network, as for example, the Three augmented network (TAN) algorithm by Friedman et al. [1997], the k-dependency Bayesian classifier (kDB) Sahami [1996] or the PC algorithm by Spirtes et al. [2000]. For many of these algorithms, is required to know and compute quantities as mutual information or cross entropy; nonetheless, since the KDE can provide an estimation of the density, is possible to estimate such quantities. In the case of the mutual information, it was estimated in the following manner:

$$\begin{aligned}
I(X_m, X_n) &= \int_{-\infty}^{\infty} f_{X_m,X_n}(x,y) \ln\left(\frac{f_{X_m,X_n}(x,y)}{f_{X_m}(x)f_{X_n}(y)}\right) dxdy \\
I(X_m, X_n) &= E_{X_m,X_n}\left[\ln\left(\frac{f_{X_m,X_n}(x,y)}{f_{X_m}(x)f_{X_n}(y)}\right)\right] \\
\hat{I}(X_m, X_n) &= \frac{1}{T+1}\sum_{t=0}^{T} \ln\left(\frac{\hat{f}_{X_m,X_n}(X_m^t, X_n^t)}{\hat{f}_{X_m}(X_m^t)\hat{f}_{X_n}(X_n^t)}\right)
\end{aligned} \tag{2.22}$$

2.2 Learning algorithms for incomplete data

The traditional and most common way to learn the parameters $\boldsymbol{\theta}$ for a probabilistic parametric model is the MLE method which can be summarized in the Eq. (2.9). However, to apply this method by traditional means, some assumptions must be done over the probability function or density of the model, i.e., the function must be at least differentiable and the evidence must be all observable or the likelihood function must be one to one with the parameter space and for all the parameters the likelihood function must has the same support Hogg and McKean [2005]. Others possible generalization of the MLE to learn the parameters are the *maximum a posteriori* (MAP) where the parameters are assumed to be random variables and they are estimated as:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{\theta}|\boldsymbol{x}^{0:T}) = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{x}^{0:T}|\boldsymbol{\theta})P(\boldsymbol{\theta}), \tag{2.23}$$

where the prior $P(\boldsymbol{\theta})$ must be defined and hyper-parameters regarding the prior must be tuned. However, some probabilistic models can be easily described with the addition of hidden or unobserved variables, then, the MLE and MAP estimations cannot be directly applied. In the presence of hidden or unobserved variables, the expectation-maximization (EM) algorithm is the standard method to estimate the model parameters. The EM algorithm will be further explained in this section and will be the main tool to estimate parameters in this thesis and can be considered also as a generalization of the Baum-Welch algorithm Rabiner [1990]. Other technique to deal with hidden variables is the *stochastic variational Bayesian* method (SVB). But as in the case of the MAP estimation, it requires the definition of priors and tuning hyper-parameters. Also, this method is better suitable when there is not a closed formula to estimate aposteriori probabilities of the hidden variables, which for this thesis, is not the case. For a brief comparison between the MLE, MAP, EM, SVB and other parameter estimation methods, the reader is referred to Tzikas et al. [2008].

2.2.1 The expectation-maximization algorithm

The expectation-maximization (EM) algorithm Dempster et al. [1977] introduces a function $\mathcal{Q}$ which serves as lower bound of the log-likelihood of the observed variables. Assume that $\boldsymbol{X}$ are the observable variables, $\boldsymbol{H}$ are the non-observable discrete ones and $\boldsymbol{x}$ is a sample from $\boldsymbol{X}$. Such bound can be deduced as follows:

$$\begin{aligned} LL(\boldsymbol{\theta}) := \ln(P(\boldsymbol{x}|\boldsymbol{\theta})) &= \sum_{\boldsymbol{R}(\boldsymbol{H})} P(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta}') \ln(P(\boldsymbol{x}|\boldsymbol{\theta})) \\ &= \sum_{\boldsymbol{R}(\boldsymbol{H})} P(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta}') \ln\left(\frac{P(\boldsymbol{x},\boldsymbol{h}|\boldsymbol{\theta})}{P(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta})}\right) \\ &= \mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}') + \mathcal{H}(\boldsymbol{\theta}|\boldsymbol{\theta}') \end{aligned} \tag{2.24}$$

In the previous equation:

$$\mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}') := \sum_{\boldsymbol{R}(\boldsymbol{H})} P(\boldsymbol{h}|\boldsymbol{x}, \boldsymbol{\theta}') \ln(P(\boldsymbol{x}, \boldsymbol{h}|\boldsymbol{\theta})) \tag{2.25}$$

$$\mathcal{H}(\boldsymbol{\theta}|\boldsymbol{\theta}') := -\sum_{\boldsymbol{R}(\boldsymbol{H})} P(\boldsymbol{h}|\boldsymbol{x}, \boldsymbol{\theta}') \ln(P(\boldsymbol{h}|\boldsymbol{x}, \boldsymbol{\theta})), \tag{2.26}$$

$\mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}')$ is the auxiliary function which serves as lower bound of the log-likelihood function; whereas $\mathcal{H}(\boldsymbol{\theta}|\boldsymbol{\theta}')$ is the cross entropy of the parameter $\boldsymbol{\theta}$ given $\boldsymbol{\theta}'$, with $\boldsymbol{\theta}'$ being an arbitrary set of parameter. The previous formulas can be exchanged with integral sign if the variables $\boldsymbol{H}$ are continuous.

In order to optimize $\mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}')$, it is required to estimate the latent or a posteriori probabilities of the hidden variables $P(\boldsymbol{h}|\boldsymbol{x}, \boldsymbol{\theta}')$, this step is called the E-step. Then, $\mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}')$ is maximized with respect to $\boldsymbol{\theta}$, this step is called the M-step. In Dempster et al. [1977], it was shown that the iteration of these two steps, gave as result, a set of parameters $\hat{\boldsymbol{\theta}}$ that reached a local optima of the likelihood function.

Algorithm 2.2 Expectation Maximization

Input: A prior $\boldsymbol{\theta}^{(0)}$, an evidence $\boldsymbol{x}$, an error ϵ
Return: Parameter $\hat{\boldsymbol{\theta}}$
  Set $l = 0$.
  while error $> \epsilon$ **do**:
    Estimate $P(\boldsymbol{h}|\boldsymbol{x}, \boldsymbol{\theta}^{(l)})$ (E-step)
    Solve $\boldsymbol{\theta}^{(l+1)} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(l)})$ (M-Step)
    Calculate error $= LL(\boldsymbol{\theta}^{(l+1)}) - LL(\boldsymbol{\theta}^{(l)})$
    if error $< \epsilon$ **do**:
      Set $\hat{\boldsymbol{\theta}} := \boldsymbol{\theta}^{(l+1)}$
    Set $l := l + 1$
  return $\hat{\boldsymbol{\theta}}$

2.2.1.1 Learning Gaussian mixture models

As example, the EM algorithm is applied to learn the parameters of a Gaussian Mixture model. A Gaussian mixture model is a soft clustering model that assumes that the data distribution can be modeled with a convex combination of normal distributions. This model is vastly used in the literature as it will be seen in Chapter 3, and therefore is briefly explained here. Consider the next definition:

Definition 2.5. A **Gaussian mixture model** (GMM) is a clustering model which uses a hidden variable $\boldsymbol{C}^t = (C_1^t, ..., C_N^t)$ independent of itself over time, which follows a multinomial distribution. Whereas, the observable variables $\boldsymbol{X}^t$ follow a MVG distribution, are independent of themselves over time and depend on $\boldsymbol{C}^t$. Set $\boldsymbol{\Theta} = \{\vartheta_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^N$, the parameters of the model. A GMM is characterized by:

1. Marginals: $P(\boldsymbol{C}^t|\boldsymbol{\Theta}) = \prod_{i=1}^{N} \vartheta_i^{c_i^t}, \quad c_i^t \in \{0,1\}, \quad \sum_{i=1}^{N} c_i^t = 1, \quad \sum_{i=1}^{N} \vartheta_i = 1$

2. Full information: $P(\boldsymbol{X}^t, \boldsymbol{C}^t|\boldsymbol{\Theta}) = \prod_{i=1}^{N} (\vartheta_i \mathcal{N}(\boldsymbol{X}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i))^{c_i^t}$

From the definition, the marginal and conditional probability of $\boldsymbol{X}^t$ can be computed:

$$P(\boldsymbol{X}^t|\boldsymbol{\Theta}) = \sum_{\boldsymbol{R}(\boldsymbol{C}^t)} P(\boldsymbol{X}^t, \boldsymbol{C}^t|\boldsymbol{\Theta}) = \sum_{i=1}^{N} \vartheta_i \mathcal{N}(\boldsymbol{X}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i). \tag{2.27}$$

$$P(\boldsymbol{X}^t|\boldsymbol{C}^t, \boldsymbol{\Theta}) = \frac{P(\boldsymbol{X}^t, \boldsymbol{C}^t|\boldsymbol{\Theta})}{P(\boldsymbol{C}^t|\boldsymbol{\Theta})} = \prod_{i=1}^{N} (\mathcal{N}(\boldsymbol{X}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i))^{c_i^t}. \tag{2.28}$$

Assuming that a prior $\boldsymbol{\Theta}^{(s)}$ has been discovered. Including the full information into Eq. (2.25), it follows:

$$\mathcal{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(s)}) = \sum_{\boldsymbol{R}(\boldsymbol{C}^{0:T})} P(\boldsymbol{c}^{0:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\Theta}^0) \left(\sum_{i=1}^{N} c_i^t \ln(\vartheta_i \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)) \right). \tag{2.29}$$

But, using the fact that $\boldsymbol{C}^t$ is independent of itself over time and marginalizing, it holds:

$$\mathcal{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(s)}) = \sum_{t=0}^{T} \sum_{i=1}^{N} \zeta^t(i) \ln(\vartheta_i \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)). \tag{2.30}$$

In the previous equation $\zeta^t(i) = P(C_i^t = 1|\boldsymbol{x}^t, \boldsymbol{\Theta}^{(s)})$. The estimation of $\zeta^t(i)$ is equivalent to perform the E-step. A closed formula is obtained:

$$\zeta^t(i) = \frac{P(C_i^t = 1, \boldsymbol{x}^t|\boldsymbol{\Theta}^{(s)})}{P(\boldsymbol{x}^t|\boldsymbol{\Theta}^{(s)})} = \frac{\vartheta_i \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{j=1}^{N} \vartheta_j \mathcal{N}(\boldsymbol{x}^t|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{2.31}$$

To perform the M-step, Eq. (2.30) has to be optimized with respect to $\boldsymbol{\Theta}$. However, there are restrictions on some parameters of the model ($\sum_{i=1} \vartheta_i = 1$) and hence a Lagrangian function $\mathcal{L}$ must be constructed to take into consideration these constraints. Then, the function $\mathcal{L}$ is derived with respect to each parameter and equalized to zero.

From the M-step, the following updating formulas are obtained:

$$\begin{aligned}
\vartheta_i^{(s+1)} &= \frac{\sum_{t=0}^{T} \zeta^t(i)}{T+1} \\
\boldsymbol{\mu}_i^{(s+1)} &= \frac{\sum_{t=0}^{T} \boldsymbol{x}^t \zeta^t(i)}{\sum_{t=0}^{T} \zeta^t(i)} \\
\boldsymbol{\Sigma}_i^{(s+1)} &= \frac{\sum_{t=0}^{T} (\boldsymbol{x}^t - \boldsymbol{\mu}_i)^\top (\boldsymbol{x}^t - \boldsymbol{\mu}_i) \zeta^t(i)}{\sum_{t=0}^{T} \zeta^t(i)}
\end{aligned} \tag{2.32}$$

2.2.2 The structural EM algorithm

In many scenarios, when it is desired to apply Bayesian networks to represent the data, no model from domain-knowledge can be provided, and discover conditional independencies for several tuples of variables can be expensive. Nevertheless, statistical methodologies can be used to discover associations in data such as the TAN , the kDB or the PC algorithm. However, in the case of the TAN or the kDB, a class variable is required, which in the case of HMM is not valid, and in the case of PC algorithm, the computation of mutual information for several variables it si unfeasible. Another possible solution is to use heuristic or meta-heuristic strategies to search the space of possible DAGs. The main of the search is to optimize the fitness of the model; nonetheless, as said in Bueno et al. [2017], when many parameters are used in dense networks, the likelihood improves but it can be due to data overfitting. Therefore, a penalized fitness can be used as objective function; some examples of such scores are the Bayesian criterion information (BIC) Schwarz [1978] or the Akaike information criterion (AIC) Akaike [1974]. Let $\#(\mathcal{B})$ be the number of parameters needed by the Bayesian network $(P, \mathcal{B})$, and T the number of instances of the evidence $\boldsymbol{x}$, then:

$$\mathrm{BIC}(\mathcal{B}, \lambda) = \ln L(\boldsymbol{x}|\mathcal{B}, \boldsymbol{\lambda}) - 0.5\#(\mathcal{B}) \ln(T+1), \tag{2.33}$$

$$\mathrm{AIC}(\mathcal{B}, \lambda) = 2\ln(L(\boldsymbol{x}|\mathcal{B}, \boldsymbol{\lambda}) - 2\#(\mathcal{B}). \tag{2.34}$$

In Friedman [1998] the structural EM (SEM) algorithm was introduced with its convergence and optimality properties. SEM helps us to find the desired model and parameters in the presence of hidden variables. The SEM takes into account the overfitting issue and during its optimization process, the score function has a BIC penalization style. In this manner, the networks are expected to be enough complex to explain the data preventing overtfitting issues. The SEM is described in Algorithm. 2.3. Friedman [1998] proposed the auxiliary function $\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^0, \boldsymbol{\lambda}^0)$ to be optimized during the structural learning of the model:

$$\mathcal{Q}(\mathcal{B}, \lambda|\mathcal{B}^0, \boldsymbol{\lambda}^0) = \mathbb{E}_{P(\boldsymbol{H}|\boldsymbol{x}, \mathcal{B}^0, \boldsymbol{\lambda}^0)}[P(\boldsymbol{x}, \boldsymbol{H}|\mathcal{B}, \boldsymbol{\lambda})] - 0.5\#(\mathcal{B}) \ln(T+1). \tag{2.35}$$

2.3 Hidden Markov models

Until now, time dynamics have been omitted in the Bayesian networks, or in other words, the samples have been assumed to be independent over time. There are Bayesian networks which address the time dependencies. Murphy [2002] discussed this issue and proposed the dynamical Bayesian networks. Two main models (and their extensions) were recognized as dynamical Bayesian networks, which are: Kalman filters and hidden Markov models. In spite of the relevance of Kalman filters, they are not going to be reviewed further in this thesis. On the other hand, the hidden Markov models are going to be reviewed in deep, since this is the base model for the extensions proposed in this thesis.

Algorithm 2.3 Structural Expectation Maximization

Input: Prior $\mathcal{B}^{(0)}$, Prior $\boldsymbol{\lambda}^{(0,0)}$, SEM error ϵ_1, EM error ϵ_2, maximum number of iterations l_{max} and data $\boldsymbol{x}$
Return: $(\boldsymbol{\lambda}^*, \mathcal{B}^*)$
  Set $l_1 = 0$
  while $l_1 < l_{max}$ **and** $\text{error}_1 > \epsilon_1$, **do**:
    Use EM with ϵ_2, $\boldsymbol{x}$ and $\boldsymbol{\lambda}^{(l_1,0)}$, find $\boldsymbol{\lambda}^{(l_1,*)}$ for graph $\mathcal{B}^{(l_1)}$
    Find a graph $\mathcal{B}^{(l_1+1)}$ that maximizes $\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l_1)}, \boldsymbol{\lambda}^{(l_1,*)})$
    Find $\boldsymbol{\lambda}^{(l_1+1,0)}$ that maximizes $\mathcal{Q}(\mathcal{B}^{(l_1+1)}, \boldsymbol{\lambda}|\mathcal{B}^{(l_1)}, \boldsymbol{\lambda}^{(l_1,*)})$
    Compute error $= \text{BIC}(\mathcal{B}^{(l_1+1)}, \boldsymbol{\lambda}^{(l_1+1,0)}) - \text{BIC}(\mathcal{B}^{(l_1)}, \boldsymbol{\lambda}^{(l_1,*)})$
    if error $< \epsilon_1$ or $l_1 \geq l_{max}$
      Use EM with ϵ_2 and $\boldsymbol{\lambda}^{(l_1+1,0)}$, find $\boldsymbol{\lambda}^{(l_1+1,*)}$ for model $\mathcal{B}^{(l_1+1)}$
      Set $\boldsymbol{\lambda}^* := \boldsymbol{\lambda}^{(l_1+1,*)}$
      Set $\mathcal{B}^* := \mathcal{B}^{(l_1+1)}$
    Set $l_1 := l_1 + 1$
  return $\boldsymbol{\lambda}^*$ and $\mathcal{B}^*$

2.3.1 Fundamentals

Let $\boldsymbol{X}^{0:T}$ be an observable stochastic process. Assume that the process $\boldsymbol{X}^{0:T}$ is dependent of a hidden or unobservable stochastic process $\boldsymbol{Q}^{0:T} = (Q^0, ..., Q^T)$, where the range of Q^t is finite, i.e, $R(Q^t) := \{1, 2, ..., N\}$, $t = 0, 1, ..., T$. These values are called states and determine the process $\boldsymbol{X}^{0:T}$. A HMM is a double stochastic process, where the stochastic hidden process $\boldsymbol{Q}^{0:T}$ is assumed to fulfill the Markov property. Whereas, $\boldsymbol{X}^{0:T}$ is usually assumed to be independent of itself over time and dependent of $\boldsymbol{Q}^{0:T}$. In a more formal way, an HMM can be defined as follows (Rabiner [1990]):

Definition 2.6. A **hidden Markov model** is a triplet $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ where $\mathbf{A} = [a_{ij}]_{i,j=1}^N$ is a matrix representing the transition probabilities between the hidden states i, j at time t, i.e. $a_{ij} = P(Q^{t+1} = j|Q^t = i)$; $\mathbf{B} = [b_j(\boldsymbol{X}^t)]_{j=1}^N$ is a vector representing the emission probability of the observations given the hidden state; if $\boldsymbol{X}^t$ is discrete and its range has κ possible values, then $b_j(\boldsymbol{X}^t) = [P(\boldsymbol{X} = \boldsymbol{x}_k|Q^t = j)]_{k=1}^\kappa$ and $\mathbf{B}$ has dimension $\kappa \times N$. If $\boldsymbol{X}^t$ is continuous, $b_j(\boldsymbol{X}^t) = f(\boldsymbol{X}^t|Q^t = j)$ and the components of $\mathbf{B}$ are pdfs. Finally, $\boldsymbol{\pi}$ is the initial distribution of hidden states $\boldsymbol{\pi} = [\pi_j]_{j=1}^N$, where $\pi_j = P(Q^0 = j)$.

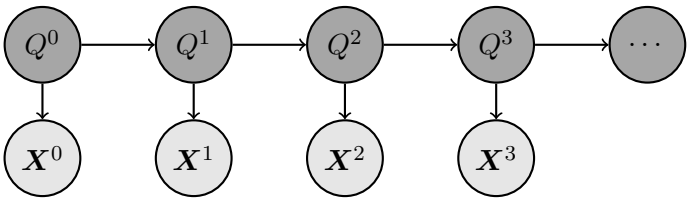


Figure 2.3: An HMM can be seen as a probabilistic graphical model.

Note that the model assumes that the transition probabilities and emission probabilities

do not depend on time t, i.e., $a_{ij}^t = a_{ij}$ and $b_j^t(\mathbf{x}^t) = b_j(\mathbf{x}^t)$. Also, an HMM can be seen as a probabilistic graphical model (Fig. 2.3), where the nodes of the graph represent random variables and the arcs represent direct probabilistic dependencies, see Murphy [2002].

Given a model $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, an instance $\boldsymbol{x}^{0:T}$ of $\boldsymbol{X}^{0:T}$ and $\boldsymbol{q}^{0:T}$ of $\boldsymbol{Q}^{0:T}$, it is possible to compute the joint probability function (or the complete information) of the model. To do this, conditional independencies can be deduced from the graph in Fig. 2.3 using D-separation (see Section 2.1.1). The found conditional independencies factorize the complete information as:

$$P(\boldsymbol{q}^{0:T}, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}) = \pi_{q^0} \prod_{t=0}^{T-1} a_{q^t,q^{t+1}} \prod_{t=0}^{T} b_{q^t}(\boldsymbol{x}^t). \tag{2.36}$$

Three main tasks can be performed in the context of HMMs. First, given a model $\boldsymbol{\lambda}$, compute the likelihood of a new instance $\boldsymbol{x}$, i.e., $P(\boldsymbol{x}|\boldsymbol{\lambda})$, which can be done using the forward-backward algorithm. Second, given a model $\boldsymbol{\lambda}$ and a set of observations $\boldsymbol{x}^{0:T}$, estimate the most probable sequence of hidden states, which can be solved using the Viterbi algorithm. Third, learn the model $\boldsymbol{\lambda}$, which is estimated with the EM algorithm instead of the maximum likelihood method. These algorithms can be further detailed in Rabiner [1990]. Nevertheless, for the sake of completeness, they are reviewed.

2.3.2 The EM algorithm for HMMs

In this section, it is shown how the EM algorithm can be used to learn the model $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$. Since this procedure will be repeatedly done during all the document, special attention to details are provided.

The algorithm consists of two steps, the expectation (E) step and the maximization (M) step. Assume that N hidden states are used and a current model $\boldsymbol{\lambda}^{(s)}$ is known. Using Eq. (2.36) in Eq. (2.25), it is obtained that $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ in the context of HMM is:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{R(\boldsymbol{Q}^{0:T})} P(\boldsymbol{q}^{0:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln \pi_{q^0} + \sum_{R(\boldsymbol{Q}^{0:T})} \sum_{t=0}^{T-1} P(\boldsymbol{q}^{0:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln a_{q^t,q^{t+1}} \\ + \sum_{R(\boldsymbol{Q}^{0:T})} \sum_{t=0}^{T} P(\boldsymbol{q}^{0:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln b_{q^t}(\boldsymbol{x}^t).\end{aligned} \tag{2.37}$$

Marginalizing the latent probabilities in each sum, $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ can be written in a tractable way:

$$\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \gamma^0(i) \ln \pi_i + \sum_{t=0}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi^t(i,j) \ln a_{ij} + \sum_{t=0}^{T} \sum_{i=1}^{N} \gamma^t(i) \ln b_i(\boldsymbol{x}^t), \tag{2.38}$$

where $\gamma^t(i) := P(Q^t = i|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)})$ and $\xi^t(i,j) = P(Q^t = i, Q^{t+1} = j|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)})$. Additionally, it is noticeable that the quantities $\gamma^t(i)$ and $\xi^t(i,j)$, $i,j = 1,...,N$, $t = 0,...,T$ can be

decomposed in factors as follows:

$$\begin{aligned}\gamma^t(i) &= P(Q^t = i|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \\ &= \frac{P(Q^t = i, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}^{(s)})}{\sum_{j=1}^N P(Q^t = j, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}^{(s)})} \\ &= \frac{P(\boldsymbol{x}^{t+1}, ..., \boldsymbol{x}^T|Q^t = i, \boldsymbol{\lambda}^{(s)})P(Q^t = i, \boldsymbol{x}^0, ..., \boldsymbol{x}^t|\boldsymbol{\lambda}^{(s)})}{\sum_{j=1}^N P(Q^t = j, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}^{(s)})} \\ &= \frac{\beta^t(i)\alpha^t(i)}{\sum_{j=1}^N \beta^t(j)\alpha^t(j)}.\end{aligned} \tag{2.39}$$

Using the same tricks, it can be obtained that:

$$\begin{aligned}\xi^t(i,j) &= P(Q^t = i, Q^{t+1} = j|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \\ &= \frac{P(Q^t = i, Q^{t+1} = j, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}^{(s)})}{\sum_{u,v=1}^N P(Q^t = u, Q^{t+1} = v, \boldsymbol{x}^{0:T}|\boldsymbol{\lambda}^{(s)})} \\ &= \frac{\alpha^t(i)a_{ij}b_j(\boldsymbol{x}^{t+1})\beta^{t+1}(j)}{\sum_{u,v=1}^N \alpha^t(u)a_{uv}b_v(\boldsymbol{x}^{t+1})\beta^{t+1}(v)}.\end{aligned} \tag{2.40}$$

$\alpha^t(i) = P(Q^t = i, \boldsymbol{x}^0, ..., \boldsymbol{x}^t|\boldsymbol{\lambda}^{(s)})$ is the forward variable and $\beta^t(i) = P(\boldsymbol{x}^{t+1}, ..., \boldsymbol{x}^T|Q^t = i, \boldsymbol{\lambda}^{(s)})$ is the backward variable. They can be estimated using the forward-backward algorithm which will be described in Section. 2.3.3. Therefore, estimating $\gamma^t(i)$ and $\xi^t(i,j)$ for $i = 1, ..., N$, $t = 0, ..., T$, in the case of HMMs corresponds to the E-step.

For the M-step, Eq. (2.38) must be optimized with respect to $\boldsymbol{\lambda}$. The updating formulas for parameters $\mathbf{A}$ and $\boldsymbol{\pi}$ can be computed using Lagrange multipliers. The restrictions are: $\sum_{i=1}^N \pi_i = 1$, $\sum_{j=1}^N a_{ij} = 1$, $i = 1, ..., N$. The Lagrangian function with multipliers $\mu_0, ..., \mu_N$ is:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + \mu_0(1 - \sum_{i=1}^N \pi_i) + \sum_{i=1}^N \mu_i(1 - \sum_{j=1}^N a_{ij}). \tag{2.41}$$

Computing the derivative of $\mathcal{L}$ in Eq. (2.41) with respect to π_i and equating to zero, it follows that:

$$\frac{\partial \mathcal{L}}{\partial \pi_i} = \frac{\gamma^0(i)}{\pi_i} - \mu_0 = 0 \tag{2.42}$$

The value of μ_0 must be determined. Using $\sum_{i=1}^N \pi_i = 1$ and $\sum_{i=1}^N \gamma^t(i) = 1$ for $t = 0, ..., T$, then:

$$\pi_i = \frac{\gamma^0(i)}{\mu_0} \Rightarrow \sum_{i=1}^N \pi_i = \frac{\sum_{i=1}^N \gamma^0(i)}{\mu_0} \Rightarrow \mu_0 = 1. \tag{2.43}$$

Therefore, the updating formula for π_i is:

$$\pi_i^{(s+1)} = \gamma^0(i), \quad i = 1, ..., N. \tag{2.44}$$

Similarly, with respect to a_{ij}:

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = \sum_{t=0}^{T-1} \frac{\xi^t(i,j)}{a_{ij}} - \mu_i = 0 \tag{2.45}$$

The value of μ_i must be also determined for $i = 1, ..., N$. Using the fact that $\sum_{j=1}^{N} a_{ij} = 1$ and $\sum_{j=1}^{N} \xi^t(i,j) = \gamma^t(i)$ for $t = 1, ..., T$, then:

$$a_{ij} = \frac{\sum_{t=0}^{T-1} \xi^t(i,j)}{\mu_i} \Rightarrow \sum_{j=1}^{N} a_{ij} = \frac{\sum_{t=0}^{T-1} \sum_{j=1}^{N} \xi^t(i,j)}{\mu_i} \Rightarrow \mu_i = \sum_{t=0}^{T-1} \gamma^t(i). \tag{2.46}$$

Therefore, the updating formula for a_{ij} is:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=0}^{T-1} \xi^t(i,j)}{\sum_{t=0}^{T-1} \gamma^t(i)}, \quad i,j = 1, ..., N. \tag{2.47}$$

The updating formula for parameter $\mathbf{B}$ relies on the assumptions made over the observable variables and the emission probabilities. Note that if the hypotheses about the transition probabilities between hidden states or observable variables change, the formulas described above do not longer hold.

2.3.3 Forward-backward algorithm

Given a current model $\boldsymbol{\lambda}^{(s)}$, to compute the forward $\alpha^t(i) = P(Q^t = i, \boldsymbol{x}^{0:t}|\boldsymbol{\lambda}^{(s)})$ and backward $\beta^t(i) = P(\boldsymbol{x}^{t+1:T}|Q^t = i, \boldsymbol{\lambda}^{(s)})$ variables they can be written in a recursive way:

$$\begin{aligned}
\alpha^{t+1}(i) &= \sum_{j=1}^{N} P(Q^t = j, Q^{t+1} = i, \boldsymbol{x}^0, ..., \boldsymbol{x}^{t+1}|\boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+1}|Q^{t+1} = i, \boldsymbol{\lambda}^{(s)}) P(Q^t = j, Q^{t+1} = i, \boldsymbol{x}^0 ..., \boldsymbol{x}^t|\boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+1}|Q^{t+1} = i, \boldsymbol{\lambda}^{(s)}) P(Q^{t+1} = i|Q^t = j, \boldsymbol{\lambda}^{(s)}) P(Q^t = j, \boldsymbol{x}^0, ..., \boldsymbol{x}^t|\boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} b_i(\boldsymbol{x}^{t+1}) a_{ji} \alpha^t(j), \quad t = 0, ..., T-1, \quad i = 1, ..., N.
\end{aligned} \tag{2.48}$$

$$
\begin{aligned}
\beta^t(i) &= \sum_{j=1}^{N} P(Q^{t+1} = j, \boldsymbol{x}^{t+1}, ..., \boldsymbol{x}^T | Q^t = i, \boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+2}, ..., \boldsymbol{x}^T | Q^{t+1} = j, \boldsymbol{\lambda}^{(s)}) P(\boldsymbol{x}^{t+1}, Q^{t+1} = j | Q^t = i, \boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+2}, .., \boldsymbol{x}^T | Q^{t+1} = j, \boldsymbol{\lambda}^{(s)}) P(\boldsymbol{x}^{t+1} | Q^{t+1} = j, \boldsymbol{\lambda}^{(s)}) P(Q^{t+1} = j | Q^t = i, \boldsymbol{\lambda}^{(s)}) \\
&= \sum_{j=1}^{N} \beta^{t+1}(j) b_j(\boldsymbol{x}^{t+1}) a_{ij}, \quad t = T-1, ..., 0, \quad i = 1, ..., N.
\end{aligned}
\tag{2.49}
$$

In summary:

$$
\alpha^{t+1}(i) = \sum_{j=1}^{N} b_i(\boldsymbol{x}^{t+1}) a_{ji} \alpha^t(j), \quad \beta^t(i) = \sum_{j=1}^{N} \beta^{t+1}(j) b_j(\boldsymbol{x}^{t+1}) a_{ij}. \tag{2.50}
$$

The initialization for the forward and backward variables are respectively $\alpha^0(i) = \pi_i b_i(\boldsymbol{x}^0)$ and $\beta^T(i) = 1$, $i = 1, ..., N$. Observe in particular that the forward variable can help us to compute the likelihood of $\boldsymbol{x}^t$ since:

$$
P(\boldsymbol{x}^{0:T} | \boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} P(\boldsymbol{x}^{0:T}, Q^T = i | \boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \alpha^T(i). \tag{2.51}
$$

However, in practice, the *scaled* forward-backward algorithm must be used since this algorithm converges to zero when the dataset is long. The long multiplications of values smaller than 1 provoke such problem. A tutorial to scale the forward-backward algorithm is presented in Rabiner [1990].

2.3.4 The Viterbi algorithm

The Viterbi algorithm (Viterbi [1967]) can be described as an MLE algorithm which uses a forward dynamic programming strategy to search optimal sequences. Given a model $\boldsymbol{\lambda}$ and a sample $\boldsymbol{x}^{0:T}$, it is desired to estimate the most probable sequence of hidden states $\boldsymbol{q}^{0:T}$. To do so, the next auxiliary problem is solved for every time instance $t = 0, ..., T$ and hidden state $i = 1, ..., N$:

$$
\delta^t(i) = \max_{q^0, ..., q^{t-1}} P(Q^t = i, \boldsymbol{x}^0, ..., \boldsymbol{x}^t, q^0, ..., q^{t-1} | \boldsymbol{\lambda}). \tag{2.52}
$$

It is noticeable that the variable $\delta^t(i)$ can be written as:

$$\begin{aligned}
\delta^t(i) &= \max_{q^0,...,q^{t-1}} \{P(\boldsymbol{x}^1, ..., \boldsymbol{x}^t, q^0, ..., q^{t-1}, Q^t = i|\boldsymbol{\lambda})\} \\
&= \max_{q^0,...,q^{t-1}} \{P(\boldsymbol{x}^t|Q^t = i, \boldsymbol{\lambda})P(\boldsymbol{x}^1, ..., \boldsymbol{x}^{t-1}, q^0, ..., q^{t-1}, Q^t = i|\boldsymbol{\lambda})\} \\
&= \max_{q^0,...,q^{t-1}} \{P(Q^t = i|q^{t-1}, \boldsymbol{\lambda})P(\boldsymbol{x}^1, ..., \boldsymbol{x}^{t-1}, q^0, ..., q^{t-1}|\boldsymbol{\lambda})\}P(\boldsymbol{x}^t|Q^t = i, \boldsymbol{\lambda}) \\
&= \max_{j=1,...,N} \{a_{ji}\delta^{t-1}(j)\}b_i(\boldsymbol{x}^t).
\end{aligned} \tag{2.53}$$

In summary:

$$\delta^t(i) = \max_{j=1,..N} \{a_{ji}\delta^{t-1}(j)\}b_i(\boldsymbol{x}^t), \tag{2.54}$$

with initialization $\delta^0(i) = \pi_i b_i(\boldsymbol{x}^0)$ for $i = 1, ..., N$. The maximization problem in Eq. (2.54) can be solved recursively as in the forward or the backward variables. However, to find the most probable sequence of states $\boldsymbol{q}^{0:T} = (q^0, ..., q^T)$ it is necessary to use an auxiliary variable $\psi^t(i)$, $i = 1, ..., N$, $t = 0, ..., T$ which will record the probabilities of transitions between states and will be used to perform a backtracking procedure. In Algorithm. 2.4 a pseudo-code of the algorithm is provided.

Algorithm 2.4 The Viterbi algorithm for HMMs

Input: Parameters $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, and a sample $\boldsymbol{x}^{0:T}$
Return: The most probable sequence of states $\boldsymbol{q}^{0:T} = (q^0, ..., q^T)$
  Compute $\delta^0(i) = \pi_i b_i(\boldsymbol{x}^0)$, $i = 1, ..., N$
  Compute $\psi^0(i) = 0$, $i = 1, ..., N$
  for $t = 1, ..., T$ **do**:
    Compute $\delta^t(i) = \max_{j=1,...,N}\{a_{ji}\delta^{t-1}(j)\}b_i(\boldsymbol{x}^t)$, $i = 1, ..., N$
    Compute $\psi^t(i) = \arg\max_{j=1,...,N}\{a_{ji}\delta^{t-1}(j)\}$, $i = 1, ..., N$
  Compute $q^T = \arg\max_{j=1,...,N}\{\delta^T(j)\}$
  for $t = T-1, ..., 0$ **do**:
    Compute $q^t = \psi^{t+1}(q^{t+1})$
  Return $\boldsymbol{q}^{0:T}$

2.3.5 Asymmetric HMMs

Asymmetric hidden Markov models were defined by Bueno et al. [2017] as a way to personalize and adapt the emission probabilities of discrete HMMs to different scenarios. This model was motivated to reduce the set of parameters in HMMs in order to avoid overfitting problems. The main idea is to take into consideration information asymmetries, i.e., the hidden variable determines a context and that context can provide a personalized emission probability for each instance of the hidden state, where the set of parameters to explain the observed data changes. To define formally an asymmetric HMM (As-HMM), it is necessary to introduce what a context-specific Bayesian network is (Boutilier et al. [1996]):

Definition 2.7. Let $\boldsymbol{X}^t = (X_1^t, ..., X_M^t)$ and Q^t be random variables. For each $q^t \in R(Q^t)$ a Bayesian network over $\boldsymbol{X}^t$ is associated, called the **context-specific Bayesian network** related to q^t or $(P_{q^t}, \mathcal{B}_{q^t})$. The following conditional distribution is defined:

$$P_{q^t}(\boldsymbol{X}^t) := P(\boldsymbol{X}^t|q^t) = \prod_{m=1}^{M} P(X_m^t|\mathbf{Pa}_{q^t}^t(X_m^t)). \tag{2.55}$$

Definition 2.8. An **asymmetric hidden Markov model** $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ is an HMM with initial distribution $\boldsymbol{\pi} = [\pi_j]_{j=1}^N$, where $\pi_j = P(q^0 = j)$, transition probabilities between the hidden states $\mathbf{A} = [a_{ij}]_{i,j=1}^N$, where $a_{ij} = P(Q^{t+1} = j|Q^t = i)$ and emission probability vector $\mathbf{B} = [b_j(\mathbf{x}^t)]_{j=1}^N$, where $b_j(\boldsymbol{x}^t) = P_j(\boldsymbol{x}^t)$ is a context-specific Bayesian network.

In Bueno et al. [2017] it is assumed that the random vector $\boldsymbol{X}^t$ is composed of discrete random variables. Therefore, the emission probabilities $\mathbf{B}$ can be seen as a list of matrices, whose dimensions will depend on the complexity of the state-specific Bayesian network. The simpler the context-specific Bayesian network (few arcs in the network), the smaller is the matrix for the emission probabilities. This model can be seen as a dynamic Bayesian network (Fig. 2.4), where the emission probabilities satisfy Eq. (2.55). In the figure, note that depending on the value of the hidden variable Q^t, the emission probability changes, given the conditional dependencies present in the state-specific Bayesian network. When $Q^t = 1$, the variable X_2^t depends on X_1^t; but, when $Q^t = 2$, both variables X_1^t and X_2^t are independent.

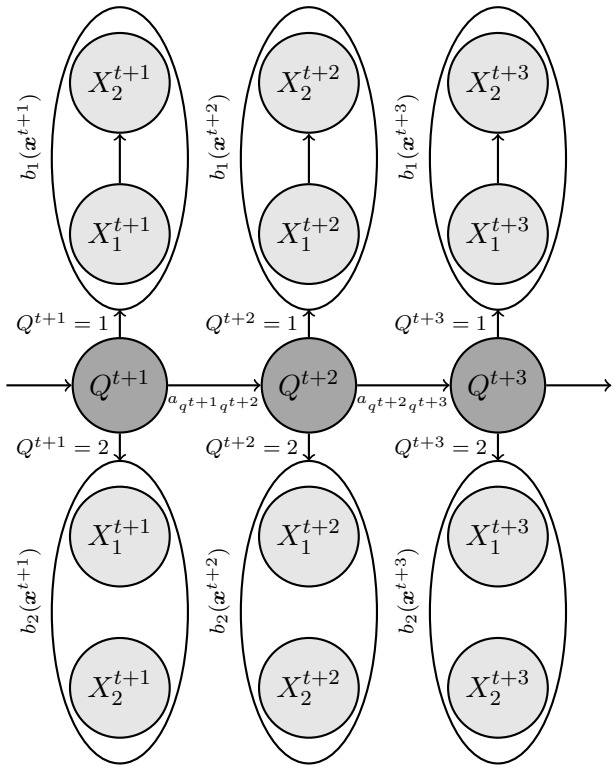


Figure 2.4: Graphical representation of an As-HMM model

2.3.5.1 EM to learn As-HMMs

The assumptions about independence between observations and states are the traditional ones. Therefore, the updating formulas for $\mathbf{A}$ and $\boldsymbol{\pi}$ are the same as in Eq. (2.44) and Eq. (2.47). On the other hand, in Bueno et al. [2017] it is assumed that each variable X_m^t has finite range and it has w_m, $m = 1, ..., M$ possible values. Also, from the definition of As-HMM, it is possible that each variable X_m^t, for each hidden state $i \in R(Q^t)$, has a different set of parents. Let κ_{im} be the number of parent configurations for variable X_m^t at the hidden state i. It is necessary to index the κ_{im} possible configurations and let $\mathbf{Pa}_i^t(X_m)_k$ be the k-th parent configuration for variable X_m^t at the hidden state i, $k = 1, ..., \kappa_{im}$. Due to the previous comments, the following restrictions hold on the emission probabilities: $\sum_{j=1}^{w_m} P(X_m = j|\mathbf{Pa}_i^t(X_m)_k) = \sum_{j=1}^{w_m} \theta_{imjk} = 1$, $i = 1, ..., N$, $m = 1, ..., M$, $k = 1, ..., \kappa_{im}$. These restrictions must be taken into account in the Lagrangian function:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + \mu_0(1 - \sum_{i=1}^{N} \pi_i) + \sum_{i=1}^{N} \mu_i(1 - \sum_{j=1}^{N} a_{ij}) + \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{k=1}^{\kappa_{im}} \nu_{imk}(1 - \sum_{j=1}^{w_m} \theta_{imjk}). \quad (2.56)$$

However the emission probabilities have changed, i.e., $b_i(\mathbf{x}^t) = P_i(\mathbf{x}^t)$, and hence the auxiliary function $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ as well:

$$\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \gamma^0(i) \ln(\pi_i) + \sum_{t=0}^{T-1} \sum_{i,j=1}^{N} \xi^t(i,j) \ln(a_{i,j}) + \sum_{t=0}^{T} \sum_{i=1}^{N} \sum_{m=1}^{M} \gamma^t(i) \ln(P(x_m^t|\mathbf{Pa}_i^t(x_m^t))). \quad (2.57)$$

If Eq. (2.56) is derived respect to θ_{imjk} and set to zero, it follows that:

$$\frac{\partial \mathcal{L}}{\partial \theta_{imjk}} = \sum_{t=0}^{T} \frac{\gamma^t(i) \chi_{(x_m^t = j, \mathbf{Pa}_i^t(x_m^t)_k)}}{\theta_{imjk}} - \nu_{imk} = 0, \quad (2.58)$$

The value of ν_{imk} must be found, to do so, note that $\sum_{j=1}^{w_m} \theta_{imjk} = 1$ and $\sum_{j=1}^{w_m} \chi_{(x_m^t = j, \mathbf{Pa}_i(x_m^t)_k)} = \chi_{(\mathbf{Pa}_i(x_m^t)_k)}$ and therefore:

$$\begin{aligned} \theta_{imjk} &= \sum_{t=0}^{T} \frac{\gamma^t(i) \chi_{(x_m^t = j, \mathbf{Pa}_i^t(x_m^t)_k)}}{\nu_{imk}}, \\ \sum_{j=1}^{w_m} \theta_{imjk} &= \frac{\sum_{t=0}^{T} \sum_{j=1}^{w_m} \gamma^t(i) \chi_{(x_m^t = j, \mathbf{Pa}_i^t(x_m^t)_k)}}{\nu_{imk}}, \\ \nu_{imk} &= \sum_{t=0}^{T} \gamma^t(i) \chi_{(\mathbf{Pa}_i^t(x_m^t)_k)}. \end{aligned} \quad (2.59)$$

Thus, the updating formula for θ_{imjk} is:

$$\theta_{imjk}^{(s+1)} = \frac{\sum_{t=0}^{T} \gamma^t(i) \chi_{(x_m^t = j, \mathbf{Pa}_i^t(x_m^t)_k)}}{\sum_{t=0}^{T} \gamma^t(i) \chi_{(\mathbf{Pa}_i^t(x_m^t)_k)}}. \quad (2.60)$$

Remember that the parameter θ_{imjk} must be updated for $i = 1, ..., N$, $m = 1, ..., M$, $j = 1, ..., w_m$ and $k = 1, ..., \kappa_{im}$.

2.3.6 Feature saliency models

In real life problems, not all the measured features can be relevant to explain a target variable, or an unsupervised behavior. Therefore, a feature subset selection (FSS)[1] procedure must be performed to choose the most relevant and non redundant features to learn a model. In this manner, noise and unrelated behaviors are omitted from the model and future analysis. As it will be seen in Section 3.2 and Section 3.4, there are different ways to perform (partially) this task. In this section, it will be reviewed a methodology to select features in probabilistic models in an embedded manner, i.e., the relevant features for the model are learned during its learning phase. Assuming that a cluster or class variable C is present, traditionally, the relevancy definition (Barddal et al. [2019]) is stated as follows:

Definition 2.9. A feature $X_m \in \text{Set}(\boldsymbol{X}_{1:M})$ is **strong relevant** if and only if, for the set $\boldsymbol{S}_m = \text{Set}(\boldsymbol{X}_{1:M}) \cap \text{Set}(\boldsymbol{X}_{1:M}) \backslash \{X_m\}$ there exist some $\boldsymbol{s}_m \in R(\boldsymbol{S}_m)$, $c \in R(C)$ and $x_m \in R(X_m)$ such that $P(x_m, \boldsymbol{s}_m) > 0$ and

$$P(c|x_m, \boldsymbol{s}_m) \neq P(c|\boldsymbol{s}_m). \tag{2.61}$$

However, for our case, since the class variable C is not present, but a cluster or hidden variable H, the relevancy definition provided by Law et al. [2004] will be used:

Definition 2.10. A feature $X_m \in \text{Set}(\boldsymbol{X}_{1:M})$ is **relevant in an unsupervised problem** if there is $x_m \in R(X_m)$ and $h \in R(H)$ such that:

$$0 < P(x_m|H = h) \neq P(x_m|H = h'), \quad \forall h' \neq h \tag{2.62}$$

Under this notion of relevancy, feature saliency models are defined. The idea behind any feature saliency model is to declare a set of binary variables, say $\{Z_m\}_{m=1}^M$, which will indicate the feature relevancy. Each Z_m variable follows a Bernoulli distribution with a parameter ρ_m, which is called the *feature saliency* of the X_m variable. If $\rho_m = 1$, it implies that the feature is relevant. If $\rho_m = 0$, it indicates that the variable is irrelevant. If $\rho_m \in (0, 1)$, a threshold $\bar{\rho}$ can be imposed as a decision boundary to determine whether or not a variable is relevant. For example, in the case of feature saliency HMMs, the feature saliency parameters are added to the emission probabilities, see Adams et al. [2016]:

$$b_i(\boldsymbol{X}^t) = \prod_{m=1}^{M} \left(\rho_m \mathcal{N}(X_m^t|\mu_{im}, \sigma_{im}^2) + (1 - \rho_m)\mathcal{N}(X_m^t|\epsilon_m, \tau_m^2)\right). \tag{2.63}$$

If $\rho_m = 1$, the pdf used for the variable X_m^t depends on the hidden state Q^t, and the variable is affected by changes in the context. This pdf will be referred to as the relevant

[1] See Saeys et al. [2007], for a review on FSS

component. Alternatively, if $\rho_m = 0$, the pdf does not depend on the hidden state Q^t and the variable X_m^t is considered as noise.

To learn the model parameters, different approaches can be used. In the case of Adams et al. [2016], the EM and MAP algorithms were used; whereas, in the case of Nguyen et al. [2015], the SVB method was applied. Further examples of feature saliency models are reviewed in Section 3.2.3.

2.4 Novelty detection

In real life problems, it is common to track the evolution of a variable over time and expect it to be in a controlled level or behave as a stationary random variable with low variance, for example, the magnitudes of vibration and temperature of a mechanical system, the amount of contaminants in the air in a city, the level of sugar in blood and so on. It is of interest to determine when there is statistical evidence to say that the observed variable has a different statistical behavior and the stationary property is no longer valid. A concept drift arises when data drifts from a statistical behavior to another one, or in other words, when there is evidence that the data has changed in distribution over time. But, sometimes, the changes in data are known and expected, and therefore the stationary property does not hold. For example, the electric profile of a machine-tool during its routine time or the climate temperature during a year. In such cases, it is interesting to determine when new trends in data appear and do not correspond to any of the known drifts. This task is known as novelty detection. The previous concepts can be put into a definition:

Definition 2.11. Let $\boldsymbol{X}^{0:T}$ be a stochastic process which is described by a sequence of distributions, densities or parameters $\boldsymbol{\theta}^{0:T}$. Then:

1. $\boldsymbol{X}^{0:T}$ is **stationary** if $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t$ for $t = 0, ..., T-1$
2. A **concept drift** at time t is characterized as $\boldsymbol{\theta}^t \neq \boldsymbol{\theta}^{t+1}$, and therefore $\boldsymbol{X}^{0:T}$ is not stationary.
3. A **novel concept** appears at t if $\boldsymbol{\theta}^t \neq \boldsymbol{\theta}^l$ for $l = 0, ..., t-1$.

As it is known, in the machine learning literature, usually, a class variable or a cluster variable C^t needs to be studied, modeled and predicted. As consequence, three kinds of concept drift can occur in the joint stochastic process: real drifts, virtual drifts and feature drifts. A real drift can be understood as a change in the conditional probability distribution of the class or clustering variables given the predictor variables. A virtual drift is seen as a modification on the joint probability distribution of the predictor variables, see Oliveira et al. [2021]. A feature drift can be understood as a change in the relevant features which define the class or clustering variables distribution, see Barddal et al. [2019]. A formal definition of these terms is provided below.

Definition 2.12. Let $(\boldsymbol{X}^{0:T}, \boldsymbol{C}^{0:T})$ be a supervised or unsupervised stochastic process, where C^t is the class or cluster variable and $\boldsymbol{X}^t$ is the predictor random vector. Then:

1. A **virtual drift** at time t happens when $P(\boldsymbol{X}^t|\boldsymbol{\theta}^t) \neq P(\boldsymbol{X}^t|\boldsymbol{\theta}^{t+1})$.
2. A **real drift** at time t happens when $P(C^t|\boldsymbol{X}^t, \boldsymbol{\theta}^t) \neq P(C^t|\boldsymbol{X}^t, \boldsymbol{\theta}^{t+1})$.
3. Let $\boldsymbol{D}^t \subset \text{Set}(\boldsymbol{X}^t)$ be the best subset of relevant features related to C^t. A **feature drift** at time t happens when $\boldsymbol{D}^t \neq \boldsymbol{D}^{t+1}$.

It is desirable to detect novel concepts and concept drifts in data. A naïve and natural strategy would be to impose a set of thresholds for the predictor variables. If any of the thresholds is violated, a concept drift is detected. However, there may be several false alarms due to outliers, and to determine optimal values for the thresholds is not a straightforward issue. A solution to these problems relies on the Page sequential test. Such test with the Chernoff bounds are used to detect outliers in stochastic processes in a robust manner, i.e., a concept drift is declared only if the number of outliers is significant.

There are other techniques to detect concept drifts, see Gama et al. [2014], such as multidimensional hypothesis testing. Nonetheless, they require labeled data and are high time consuming, which may be unfeasible for some applications.

2.4.1 The Page sequential test

The Page sequential test (Page [1954]) is a tool to detect data deviation in time series. Assume that a stochastic process $\boldsymbol{X}^{0:T}$ is being observed and let $s^t : R(\boldsymbol{X}^{0:t}) \to \mathbb{R}$ be a function which summarizes the observations $\boldsymbol{x}^{0:t}$. It is interesting to determine if $\boldsymbol{x}^t$ is anomalous given the previous data summaries values $\{\boldsymbol{s}^l(\boldsymbol{x}^{0:l-1})\}_{l=0}^{t-1}$. For that, it is imposed that $\boldsymbol{x}^t$ is anomalous in an increasing manner if, given a threshold $\Gamma > 0$, the following condition is met for any instant:

$$s^t(\boldsymbol{x}^{0:t}) \geq s^l(\boldsymbol{x}^{0:l}) + \Gamma, \tag{2.64}$$

or more simply, if $s^t(\boldsymbol{x}^{0:t})$ deviates Γ units from any previous $s^l(\boldsymbol{x}^{0:l})$, $l = 0, ..., t-1$. Then the observation $\boldsymbol{x}^t$ is considered as an outlier. The previous equation can be expressed in a more useful way:

$$s^t(\boldsymbol{x}^{0:t}) - \min_{l=0,\ldots,t-1} s^l(\boldsymbol{x}^{0:l}) \geq \Gamma. \tag{2.65}$$

The previous equation is the Page sequential test for increasing schemes. This implies that, if the summary $s^t(\boldsymbol{x}^{0:t})$ decreases with respect to $\{\boldsymbol{s}^l(\boldsymbol{x}^{0:l})\}_{l=0}^{t-1}$, it will not trigger the previous equation. The Page sequential test for decreasing schemes can be stated in a similar manner:

$$\max_{l=0,\ldots,t-1} s^l(\boldsymbol{x}^{0:l}) - s^t(\boldsymbol{x}^{0:t}) \geq \Gamma \tag{2.66}$$

In Page [1954], $\boldsymbol{x}^t$ was considered as a one dimensional increasing time series and $s^t(\boldsymbol{x}^{0:t}) = \sum_{l=0}^{t} x^l$; in other words, the test was checking if the accumulative values of x^t increased drastically over time. Another example of function s^t but for multivariate data can be found in Diaz-Rozo et al. [2020] which is an adaptation of the Page-Hinkley test proposed by Hinkley [1971]. A Gaussian mixture model was used to explain the data and the quality of the model was measured with the BIC value. They proposed a s^t function based on

the mean temporal deviation of the current BIC value from its current mean value, i.e., $s^t(\boldsymbol{x}^{0:t}) = \frac{1}{t}\sum_{l=0}^{t}(\mathrm{BIC}(\boldsymbol{x}^t) - \frac{1}{l}\sum_{k=0}^{l}\mathrm{BIC}(\boldsymbol{x}^k) + \delta)$ with $\delta > 0$. Despite its simplicity, the Page test is a stronger and faster tool to deal and detect outliers in sequential data.

2.4.2 The Chernoff bounds

The identification of an anomaly can provoke false positive alarms. Therefore, it is necessary to determine the amount of anomalies in order to generate an alarm of undesirable or unknown behavior. The Chernoff bounds, see Chernoff [1952], can be used for this purpose, as in Diaz-Rozo et al. [2020], applied to detect anomalous behavior in industrial environments. These bounds can determine an estimation of the minimum amount of anomalies in a window time to declare a change in the data trend. To determine the number of anomalies, recall the sampling problem: given an error ϵ, a confidence γ and a target proportion p of anomalies, the sample size n^*, such that the sampled proportion p_{n^*} fulfills $p_{n^*} < p$, must be estimated. In probability terms, this problem can be stated as:

$$P(p - p_{n^*} < \epsilon) \geq 1 - \gamma, \tag{2.67}$$

The Markov inequality can be applied, and take an arbitrary $l > 0$ to obtain that:

$$\begin{aligned} P(n^* p_{n^*} > n^*(p-\epsilon)) &= P\left(e^{ln^* p_{n^*}} > e^{ln^*(p-\epsilon)}\right) \\ &< E\left[e^{lnp_{n^*}}\right] e^{-ln(\epsilon-p)} \\ &\leq \inf_l E\left[e^{lnp_{n^*}}\right] e^{-ln(p-\epsilon)}, \end{aligned} \tag{2.68}$$

It is required to compute $\inf_l E\left[e^{ln^* p_{n^*}}\right] e^{-ln^*(p-\epsilon)}$. Observe that $\mathbb{E}\left[e^{ln^* p_{n^*}}\right]$ is the moment-generating function of a binomial distribution i.e., $\mathbb{E}\left[e^{\ln^* p_{n^*}}\right] = (1 - p + pe^l)^{n^*}$. Define the auxiliary function $f(l) := \ln\left(\mathbb{E}\left[e^{\ln^* p_{n^*}}\right] e^{-\ln^*(p-\epsilon)}\right) = n^* \ln(1-p+pe^l) - \ln^*(p-\epsilon)$, its optimal value is $l^* = \ln\left(\frac{(p-\epsilon)(1-p)}{(1-p+\epsilon)p}\right)$. Substituting l^* in $e^{f(l)}$, it follows that:

$$\begin{aligned} \inf_l \mathbb{E}\left[e^{ln^* p_{n^*}}\right] e^{-ln^*(p-\epsilon)} &= \left(\left(\frac{1-p}{1-p+\epsilon}\right)^{1-p+\epsilon}\left(\frac{p}{p-\epsilon}\right)^{p-\epsilon}\right)^{n^*} \\ &= e^{-D_{KL}(p-\epsilon||p)n^*}, \end{aligned} \tag{2.69}$$

where $D_{KL}(p-\epsilon||p)$ is the divergence of Kullback-Leibler for two Bernoulli distributions with parameters $p-\epsilon$ and p. Substituting this result in Eq. (2.67), the following inequality holds:

$$\begin{aligned} e^{-D_{KL}(p-\epsilon||p)n^*} &\geq 1 - \gamma \\ n^* &\geq \frac{-\ln(1-\gamma)}{D_{KL}(p-\epsilon||p)} \end{aligned} \tag{2.70}$$

The previous equation states that if a sample of at least $\frac{-\ln(1-\gamma)}{D_{KL}(p-\epsilon||p)}$ instances is taken, with a confidence of γ and true proportion of anomalies p; then the estimated proportion p_{n^*},

has an error of ϵ with respect to p. If the estimated proportion p_{n^*} overpasses p, there is evidence that the real proportion of anomalies is greater than p and an alarm of unknown or undesirable behavior must be generated, i.e., a concept drift.

2.5 Ball bearings

2.5.1 Ball bearing fundamental frequencies

One of the common practical application of this thesis is focused on determining the health state of ball bearings. Therefore, it must be known what a ball bearing is and its fundamental frequencies. A ball bearing is a machine component that is usually used to allow a shaft to rotate in int longitudinal axis. A ball bearing is composed of many components; namely, outer race, inner race, rollers and cage. Fig. 2.5 shows how a bearing looks like.

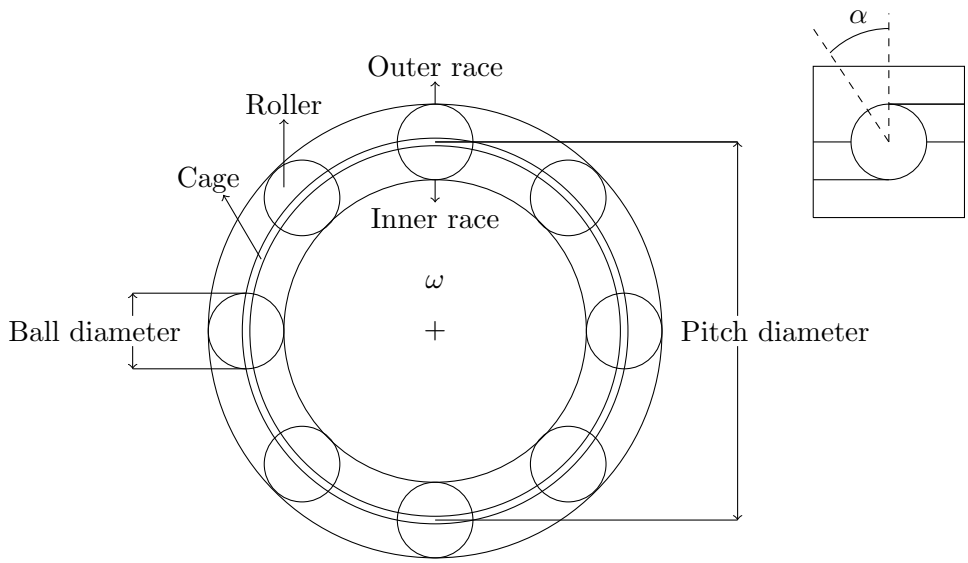


Figure 2.5: Bearings schematic figure. In this case the bearing is rotating counterclockwise at a rotational speed of ω

If the geometry of the ball bearing is known, its fault frequencies can be computed (Graney and Starry [2011]):

$$\begin{aligned}
\text{BPFO} &= \frac{r\omega}{2}(1 - \frac{B}{P}\cos(\alpha)),\\
\text{BPFI} &= \frac{r\omega}{2}(1 + \frac{B}{P}\cos(\alpha)),\\
\text{FTF} &= \frac{\omega}{2}(1 - \frac{B}{P}\cos(\alpha)),\\
\text{BSF} &= \frac{P\omega}{2B}(1 + (\frac{B}{P}\cos(\alpha))^2),
\end{aligned} \tag{2.71}$$

where BPFO stands for ball pass frequency outer race, BPFI is ball pass frequency inner

race, FTF is fundamental train frequency, BSF is ball spin frequency, r is the number of rollers, ω is the shaft frequency measured in Hz, B is the ball diameter and is measured in mm, P is the pitch diameter and is measured in mm, and finally α is the contact angle of the rollers with the races with respect to the perpendicular line. If the previous frequencies are observed with a high magnitude in the taken samples, it is expected that these components are not acting normally or are degrading.

2.5.2 Signal processing and feature extraction

It is known which frequencies must be monitored in ball bearing surveillance. However it must be declared how to measure them. When a sensor (e.g., accelerometer, thermocouple) is used to measure a physical event, the output is a signal that carries relevant information. At first glance, this raw signal may not provide any insight about what is happening, and therefore, features from the underlying physical phenomenon have to be extracted.

Suppose a signal $f(t)$ with period $\mathcal{T} \in \mathbb{R}$ has been measured. This signal is composed of several frequency components that contain important information. However, those frequency components are not explicit in $f(t)$. The Fourier transform is commonly used to perform the decomposition of the signal in frequency components. Eq. (2.72) defines the Fourier transform $\hat{f}(z)$ of the signal[2]:

$$\hat{f}(z) = \int_{-\mathcal{T}/2}^{\mathcal{T}/2} f(t) e^{-2\pi i z t} dt. \tag{2.72}$$

Assume that $f(t)$ satisfies the following condition $\int_{-\frac{\mathcal{T}}{2}}^{\frac{\mathcal{T}}{2}} |f(t)|^2 dt < \infty$. In such case, $f(t)$ belongs to a vector space called $L^2\left[-\frac{\mathcal{T}}{2}, \frac{\mathcal{T}}{2}\right]$ or the square-integrable functions in the interval $\left[-\frac{\mathcal{T}}{2}, \frac{\mathcal{T}}{2}\right]$. This space has a "base" $E = \{e^{2\pi i \frac{n}{\mathcal{T}} t}\}_{n \in \mathbb{Z}}$, which can generate any vector $g \in L^2\left[-\frac{\mathcal{T}}{2}, \frac{\mathcal{T}}{2}\right]$. Therefore, $f(t)$ can be generated by means of linear combinations (see Rudin [1976]) of complex numbers $\{c_z\}_{z \in \mathbb{Z}}$:

$$f(t) = \sum_{n \in \mathbb{Z}} c_n e^{2i\pi \frac{n}{\mathcal{T}} t}, . \tag{2.73}$$

The righthand side of Eq. (2.73) is known as the Fourier series of $f(t)$. On the other hand, Euler's formula indicates that $e^{i\theta} = \sin(\theta) + i\cos(\theta)$, which implies that $f(t)$ can be decomposed into periodic functions. The coefficients $\{c_n\}$, $n \in \mathbb{Z}$ denote which periodic functions or frequencies play a more important role in $f(t)$. Observe also that the function $f(t)$ can be reconstructed if the coefficients $\{c_n\}_{n \in \mathbb{Z}}$ are known.

The space $L^2\left[-\frac{\mathcal{T}}{2}, \frac{\mathcal{T}}{2}\right]$ has an inner product which is[3] $\langle f, g \rangle = \frac{1}{\mathcal{T}} \int_{-\mathcal{T}/2}^{\mathcal{T}/2} f(t)\bar{g}(t) dt$. According to this inner product, any two different vectors $f, g \in E$ with $f \neq g$, have the following properties: $\langle f, g \rangle = 0$ and $\langle f, f \rangle = 1$. It follows from the above properties that each coefficient

[2] i denotes the complex number that solves $x^2 + 1 = 0$ or simply $i = \sqrt{(-1)}$

[3] Here $\bar{g}(t)$ represents the complex conjugation of $g(t)$

c_n can be expressed as:

$$c_n = \langle f(t), e^{2i\pi \frac{n}{\mathcal{T}} t} \rangle = \frac{1}{\mathcal{T}} \int_{-\mathcal{T}/2}^{\mathcal{T}/2} f(t) e^{-2i\pi \frac{n}{\mathcal{T}} t} dt. \tag{2.74}$$

Note that c_n can be computed as the Fourier transform of $f(t)$ evaluated at $\frac{n}{\mathcal{T}}$, i.e., $c_n = \frac{1}{\mathcal{T}} \hat{f}(\frac{n}{\mathcal{T}})$. The magnitude of the frequency $z_n := n/\mathcal{T}$ for the signal $f(t)$ is the value $|c_n|$.

Recall that a ball bearing is a rotating machine element. Any crack in a ball bearing component will cause that the failure frequencies appear in the signal, namely FTF, BPFI, BPFO, and BSF. The signal spectrum (the map $z_n \mapsto |c_n|$) can be estimated with the use of the Fourier transform. But, noise can be present in the sample. Therefore, the signal must be filtered to extract the magnitude of the failure frequencies.

A naive way to filter the signal, such that the spectrum only lies between the frequencies $[z_0, z_1]$, is to multiply the spectrum by the function $\chi_{[z_0,z_1]}$. In this manner, only the spectral magnitudes between $[z_0, z_1]$ are preserved. However, if the inverse Fourier transform on the filtered spectrum is used, it can be noted that an infinite impulse response in $f(t)$ will appear on all the time domain, which will generate inconsistent time behaviors into $f(t)$. To alleviate this issue, a better filter function on the spectrum domain must be selected, such that it generates a finite impulse response in time in $f(t)$. In Alexander and Sadiku [2007], a deeper and more complete discussion about analogical and digital filtering can be found.

For this thesis a digital filter is designed with the use of spectral kurtosis. When a periodic transient force is detected in a signal, the kurtosis value increases as observed by Antoni [2007]. Then, it is desired to find the optimal filter which highlights the highest kurtosis. Some traditional strategies like the fast kurtogram (see Antoni [2007]) can be used to design this optimal filter. Other signal processes as Hilbert transform[4] can be used additionally to extract the envelope of the signals (Bechhoefer [2005]). The envelope is generally generated by the ball bearing cracks.

In this thesis, to extract the magnitudes of the BPFO, BPFI, FTF and BSF, from the signal spectrum, an optimal filter based on kurtosis and envelope extraction is used as in Wang and Liang [2010].

2.5.3 Yule-Walker equations

The Yule-Walker equations Box and Jenkins [1976] is a methodology to determine the autoregressive (AR) order of a random variable. A linear AR process with k time lag coefficients for a one-dimensional variable X^t can be described as:

$$X^t = \phi_{k1} X^{t-1} + \cdots + \phi_{kk} X^{t-k} + \epsilon^t, \tag{2.75}$$

where $\epsilon^t \sim \mathcal{N}(0, \sigma^2)$ is an error term following a Gaussian distribution with mean zero and variance σ^2. The correlogram function ρ_k returns the correlation between X^t and X^{t-k}.

[4]The Hilbert transform can be stated as: $H[f](z) = \frac{1}{\pi} \text{p.v} \int_{-\infty}^{\infty} \frac{f(t)}{z-t} dt$, where p.v. is the Cauchy principal value

Define $\bar{X}^t := X^t - \mu_X$ where μ_X is the mean of X^t and $\zeta_k := E[\bar{X}^t \bar{X}^{t-k}]$, which is the expected value of the product of both shifted variables. The correlogram function is computed as:

$$\rho_k := \frac{\zeta_k}{\zeta_0}.$$

The partial correlogram function $\Phi(k)$ encodes the correlation between variables X^t and X^{t-k} once the effect from intermediary lags has been removed. To determine these partial correlations, observe that for $l \in \{1, ..., k\}$:

$$\begin{aligned} \bar{X}^t &= \phi_{k1}\bar{X}^{t-1} + \cdots + \phi_{kk}\bar{X}^{t-k} + \epsilon^t \\ \bar{X}^t\bar{X}^{t-l} &= \phi_{k1}\bar{X}^{t-1}\bar{X}^{t-l} + \cdots + \phi_{kk}\bar{X}^{t-k}\bar{X}^{t-l} + \epsilon^t\bar{X}^{t-l} \\ \rho_l &= \phi_{k1}\rho_{l-1} + \cdots + \phi_{kk}\rho_{k-l} \end{aligned} \tag{2.76}$$

In the last line of Eq. (2.76), it was applied the expectation operator and divided by ζ_0. It was assumed that $\mathbb{E}[\bar{X}^{t-l}\epsilon^t] = 0$ for all t, which implies that X^t is not correlated with the error term; a plausible hypothesis in real situations. Additionally, $\rho_0 = 1$. Moreover, notice that if these equations are computed for $l = 1, ..., k$, a system of linear equations is obtained, which corresponds to the Yule-Walker equations:

$$\begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_k \end{bmatrix} = \begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \phi_{k1} \\ \phi_{k2} \\ \vdots \\ \phi_{kk} \end{bmatrix}.$$

The partial correlogram function returns $\Phi(k) := \phi_{kk}$. Note that if it desired to evaluate up to k lags for the partial correlogram function, k linear systems must be constructed and solved.

Assume that the sample is white noise. Then the parameter ϕ_{kk} is distributed approximately as $\mathcal{N}(0, 1/T)$. With this information, it is possible to perform hypothesis testing to determine the relevancy of each lag parameter. If $\Phi(p^*)$ is the higher time lag coefficient that is significantly different from zero, then p^* is considered the AR order of the model (Box and Jenkins [1976]).

Chapter 3

State of the art

In this chapter, traditional and recent articles regarding different areas related to HMMs are reviewed. As stated in Chapter 1, this thesis makes contributions to HMMs in the following topics:

1. As-HMMs with Gaussian and autoregressive (AR) properties in the observable variables.

2. Embedded feature selection in As-HMMs.

3. Online ball bearing prognosis using As-HMMs and machine learning.

4. Online dynamic embedded feature selection in As-HMMs.

5. KDE for non-Gaussian estimations in As-HMMs.

Therefore, the state of the art is divided into five sections. Section 3.1 is related mostly to AR models with symmetric and asymmetric HMMs and machine learning models. Section 3.2 discusses FSS techniques related to HMMs and other machine learning models, making focus on embedded strategies. Section 3.3 reviews articles related to online prognosis for industrial assets. As discussed in Chapter 1, it is possible that the available datasets for prognosis are imbalanced; then, the state of the art is separated into three categories: methodologies which use previous run to failure (RTF) data to predict, which use RTF times to predict, and which assume no previous RTF data or times. Section 3.4, reviews feature selection methodologies for online analysis; the articles are separated into filter, wrapper and embedded methodologies. Finally, Section 3.5 presents how Bayesian networks and HMMs have been mixed with KDEs. From all the reviewed articles, some of them are selected to be compared in the contributions part (Part II). The selected articles are mentioned in their corresponding chapter.

3.1 Offline models

The first step taken in this thesis was to make an exhaustive review of AR-HMMs and asymmetric models, in order to determine if the proposed HMM extensions and algorithms have not been previously introduced. In this section, several variants of HMMs are reviewed. As a general observation, it was noted that most of the reviewed HMMs assumed fully dependent or independent variables, which can provoke the introduction of several or few parameters, which, at the same time, can cause overfitting or model under-specification, respectively. Then, As-HMMs are an intermediate solution to tackle these issues. The reviewed articles are tangled with the contributions in Chapter 4 and Chapter 5. The results obtained in these sections can also be found in Puerto-Santana et al. [2018] and Puerto-Santana et al. [2022b]. In Table 3.1 the reviewed articles are listed.

3.1.1 Symmetric HMMS

3.1.1.1 Auto-regressive HMMs

One of the first combinations of HMMs and AR models attempted to process speech data was introduced in Poritz [1982]. The author added autoregressive polynomials to the Gaussian emission probabilities. The coefficients were determined via the Baum-Welch algorithm (Rabiner [1990]). Later, Juang and Rabiner [1985] introduced AR mixtures of Gaussian hidden Markov models (AR-MoG-HMMs) where the emission probabilities were modelled as mixtures of Gaussians. Then, Kenny et al. [1990] proposed a vectorial AR multivariate Gaussian HMM (VAR-MVGHMM). This model enabled variables to have temporal dependencies with all the other variables. Again, the model was used for speech recognition. More recently, Bryan and Levinson [2015] modified the emission probabilities such that they behaved as an AR Gaussian but with an error coefficient given by the linear prediction residuals (Itakura [1975]).

Others also considered variations of HMMs such as hidden semi-Markov models (HSMMs), where the time duration of each hidden state can be modified to not always follow a geometric distribution, or hierarchical hidden Markov models (HHMMs), where AR variables were added. For instance, Nakamura et al. [2015] proposed an AR-HSMM, where AR variables and non-AR variables could be considered in the same model depending on the modeler's decision. Malesevic et al. [2018] proposed a vector AR hierarchical hidden semi-Markov model (VAR-HHSMM) to classify and determine hand movements.

Other approximations of HMMs with AR properties can be found in Hamilton [1989] and Hamilton [1990]. The author proposed an edited log-likelihood function to represent the AR behavior in data. As a result, Markov mean-switching AR models (MMSAR) and linear Markov-switching AR model (LMSAR) were proposed and their parameters were calculated with the EM algorithm. Next, Cheng [2016] proposed the transitional Markov switching autoregressive (TMSAR) model as an extension of MMSAR and LMSAR models. In this case, the emission probabilities depended on past values of the hidden process in order to

Topic	Name	Cite
Symmetric HMMs	**Auto-regressive HMMs**	
	AR polynomials	Poritz [1982]
	AR-MoG-HMM	Juang and Rabiner [1985]
	MMSAR	Hamilton [1989]
	VAR-MVGHMM	Kenny et al. [1990]
	LMSAR	Hamilton [1990]
	Linear error coefficient	Bryan and Levinson [2015]
	AR-HSMM	Nakamura et al. [2015]
	TMSAR	Cheng [2016]
	VAR-HHSMM	Malesevic et al. [2018]
	Modified hidden variables	
	Memoryless hidden variables	Asahara et al. [2012]
	AR-HO-HMM	Seifert et al. [2014]
	Missing data in HMMs:	
	Missing at random assumption	Stanculescu et al. [2014]
	Missing data as latent variables	Dang et al. [2016]
Asymmetric models	Similarity networks	Heckerman [1990]
	Bayesian multinets	Geiger and Heckerman [1996]
	Context-specific Bayesian networks	Boutilier et al. [1996]
	BMM	Bilmes [2003]
	Chow-Liu trees in HMMs	Kirshner et al. [2004]
	CEG	Smith and Anderson [2008]
	MRF-HMMs	Stadler and Mukherjee [2013]
	SGM	Nyman et al. [2014]
	Dynamic chain events graph	Barclay et al. [2015]
	As-HMM	Bueno et al. [2017]
	AsLG-HMM	Puerto-Santana et al. [2018]
	AR-AsLG-HMM	Puerto-Santana et al. [2022b]

Table 3.1: Reviewed articles about asymmetric HMMs and auto-regressive HMMs

determine changes in the model mean and weights. The authors used maximum likelihood methods with a Newton-Raphson strategy to estimate the model parameters.

3.1.1.2 HMMs that modify the hidden variables

In more recent works, new approaches have been proposed in which the assumptions about the hidden variables that govern the process were modified, such as the model given by Asahara et al. [2012], where the authors edited an AR hidden Markov model (AR-HMM) by introducing a memoryless hidden variable. The Markovian hidden states had a probabilistic dependency on this memoryless hidden variable. Later, AR higher-order HMMs (AR-HO-HMMs) were introduced in Seifert et al. [2014]; the authors not only considered an autoregressive property in the observations, but also a fixed order Markov assumption in the hidden states specified by the user. They used mixtures of Gaussians with AR properties for the emission probabilities.

3.1.1.3 Missing data in HMMs

Other works focused on missing data. In Stanculescu et al. [2014], an AR-HMM with a missing at random assumption was proposed to perform exact inference in such scenarios. In Dang et al. [2016] the missing data was considered as latent variables; when the sampling rate of a target signal was not high enough, hidden variables were added between observations. Additionally, the authors proposed a modified forward-backward algorithm and updating formulas for the Baum-Welch algorithm.

3.1.2 Asymmetric HMMs

Asymmetric models as defined in Section 2.3.5 can be viewed as probabilistic models where, depending on the instance of a context variable, the probabilistic relationships among the observable variables may change.

Regarded as asymmetric probabilistic graphical models, the Bayesian multinets introduced in Geiger and Heckerman [1996] were used to describe different local graphical models depending on the values of certain observed variables. Previously, in the area of decision making, similarity networks introduced by Heckerman [1990], allowed the creation of independent influence diagrams[1]. In parallel to these works, context-specific independence in Bayesian networks in Boutilier et al. [1996] were introduced. The authors used tree structured conditional probability distributions with a D-separation-based algorithm to determine statistical dependencies between variables according to contexts given by instantiations of subsets of variables. Following these ideas, more recently in Nyman et al. [2014], stratified graphical models (SGMs) were proposed, where the concept of stratum was introduced to allow different factorizations for a probability distribution depending on the values of some of the variables. The authors proposed a nonreversible Metropolis-Hastings algorithm to calculate marginal likelihoods and learn decomposable SGMs. Other approximation was proposed by Smith and Anderson [2008] who introduced chain events graphs (CEGs). A CEG consists of a directed colored graph obtained from a staged tree[2] by successive edge contraction operations. The obtained graphical model can represent conditional independence and causal behavior that traditional Bayesian networks cannot show. Later, a dynamic version was proposed (Barclay et al. [2015]).

Other authors have attempted to combine asymmetric models with HMMs. For example, in Bilmes [2003] the buried Markov models (BMMs) were introduced. In this article, the models of Kenny et al. [1990] were used, but the temporary dependencies could vary depending on the hidden state. These context-specific dependencies were learned using mutual information strategies. Kirshner et al. [2004] used Chow-Liu trees and conditional Chow-Liu trees coupled with HMMs. The HMMs were used to model the dynamic behavior of a process,

[1] An influence diagram is a probabilistic graphical model used for decision problems, where random, decision and value nodes are present (Shachter [1988])

[2] A staged tree is a probabilistic graphical model, where the graph is a tree and the nodes are random variables whose non leaf variables are identified with the same color if they have the same conditional probabilistic relationships with their children nodes (Smith and Anderson [2008])

and the Chow-Liu trees were used to model the emission probabilities. A Chow-Liu tree or conditional Chow-Liu tree was associated with each value of the hidden variable. The parameters of the model were computed with the EM algorithm; specifically, the tree structure was determined in the maximization step. However, the model was specified only for discrete variables. In Stadler and Mukherjee [2013], the authors proposed a learning algorithm based on the EM algorithm to generate sparse precision matrices, i.e., each hidden state had its own sparse precision matrix which could be interpreted as a Markov random field (MRF)[3]. More recently, asymmetric hidden Markov models (As-HMMs) were proposed in Bueno et al. [2017], where a local graphical model was associated with each value of the hidden variable, and the graphical model was not restricted to Chow-Liu trees. However, again only models with discrete observable variables were allowed. Puerto-Santana et al. [2018] alleviated this issue with the asymmetric linear Gaussian HMMs (AsLG-HMMs), where the emission probabilities were modeled as conditional linear Gaussian Bayesian networks. This model will be further explained in Chapter 4. Finally, the previous model was further generalized in Puerto-Santana et al. [2022b], where an asymmetric HMM for continuous variables was proposed, which was capable of determining the AR order for each variable depending on the hidden state (AR-AsLG-HMMs). This model will be detailed in Chapter 5.

3.2 Offline feature subset selection

Regarding FSS, the state-of-the-art strategies are usually grouped into three categories: filter, wrapper and embedded. The filter techniques try to determine the relevant features depending on intrinsic data characteristics such as entropy, variance, information, etc. The wrapper techniques depend on a machine learning algorithm and its performance. Subsets of features are generated and a model is learned for each subset. The subset with the best performance is selected. Embedded unsupervised techniques determine the relevant features during the learning procedure of the model. There are also dimensionality reduction strategies as in Jolliffe [1986] or Chang et al. [2016]. In these articles, the features are projected into lower dimensionality spaces where the information is concentrated and these new features are used as predictors. Nevertheless, this kind of strategies are beyond the scope of this thesis, since model interpretation is hard to handle. The reviewed articles in this section are the basis for Chapter 6. The results of that chapter were exposed in Puerto-Santana et al. [2022c], where an embedded FSS methodology for As-HMMs is proposed. In Table 3.2 the reviewed articles are listed.

3.2.1 Filter techniques

3.2.1.1 Supervised

Among the traditional filter FSS techniques, the correlation based feature selection (CFS)

[3]A Markov random field is a probabilistic graphical model which represents a set of variables which follow the Markov property. The graph must be undirected and may have cycles.

Topic	Name	Cite
Filter	**Supervised**	
	CFS	Hall et al. [1999]
	RELIEF	Kira et al. [1992]
	Markov blanket and HMM	Zhou et al. [2017]
	AdaBoost and HMM	Lv and Nevatia [2006]
	Ensemble of filters and HMM	Momenzadeh et al. [2019]
	Unsupervised	
	RELIEF for clustering	Dash and Ong [2011]
Wrapper	A greedy backward FSS with HMM classifier	Yue et al. [2015]
	PSO with HMM classifier	Farag et al. [2016]
Embedded	**Supervised**	
	Lasso regression	Tibshirani [1996]
	Penalized neural networks	Setiono and Liu [1997]
	Reinforcement learning in neural networks	Mnih et al. [2014]
	Clustering	
	GMM-FS	Law et al. [2004], Zhu et al. [2012]
	GMM-LFS	Li et al. [2009]
	Semi-supervised GMM-LFS	Guerra et al. [2014]
	VB-SMM-LFS	Nguyen et al. [2015]
	iGMM-LFS	Song et al. [2021]
	HMMs	
	FS-HMM	Adams et al. [2016]
	SHMM-LFS	Zheng et al. [2018]
	Discrete FS-HMM	Adams and Beling [2020]
	FS-AsHMM	Puerto-Santana et al. [2022c]

Table 3.2: Table of reviewed articles for offline feature selection

(Hall et al. [1999]) is found. CFS was designed to search for a subset of features which maximizes the feature relevancy with respect to a class variable and minimizes the redundancy between them. RELIEF (Kira et al. [1992]) provided a relevancy score for each feature based on distances between classes. Features whose relevancy overpassed a threshold were selected.

As regards filter algorithms with supervised data related to HMMs, the following articles were found: Zhou et al. [2017] proposed a sequential data feature selection algorithm based on Markov blankets. Their methodology gradually computed the Markov blanket of a target or class variable using the HITON algorithm (Alifers et al. [2003]). The HITON algorithm was fed with an HMM to learn the corresponding Markov blanket. The variables in the Markov blanket of the class variable were used as predictor features of a classification model. In Lv and Nevatia [2006], for each class value and variable, an HMM, which was used as a classifier, was learned. Next, an AdaBoost algorithm was employed to select which HMMs improved the accuracy of the prediction. Momenzadeh et al. [2019] coupled discrete HMMs with different feature selection filtering scores. An HMM was created using the ranking information obtained by the filters. The resultant emission probabilities were used as a relevancy score.

3.2.1.2 Unsupervised

Only one filter strategy for variable selection was found in the case of unsupervised data. Dash and Ong [2011] used the RELIEF algorithm (Kira et al. [1992]) to discriminate features. The authors used the K-medoid algorithm to generate artificial class labels and execute the RELIEF algorithm using them. This process was iterated as many times as the user determined.

3.2.2 Wrapper techniques

In wrapper techniques, few works were found, all of which look for the best set of variables to improve the score of a clustering model. For instance, Yue et al. [2015] used a greedy-backward FSS algorithm to select the features for an adaptive variable duration mixture of Gaussian HMMs. Farag et al. [2016] applied the particle swarm optimization (PSO) algorithm (Reynolds [1987]) to maximize the HMM accuracy. In both cases, heuristic or meta-heuristic methods were employed to find the best set of features for a HMM classifier.

3.2.3 Embedded techniques

3.2.3.1 Supervised

Some embedded techniques for supervised problems are reviewed. For example, the well-known work of Tibshirani [1996], where the lasso regression was introduced. Setiono and Liu [1997], added a regularization term was in the learning phase of neural networks in order to determine relevant features. More recently, Mnih et al. [2014] introduced a convolutional neural network capable of selecting image sections to be processed, but the classifier had to be learned using reinforcement learning.

3.2.3.2 Clustering

In Law et al. [2004], the concept of feature saliency was introduced and applied to perform model learning and feature selection simultaneously in clustering models. The feature saliencies were utilized to indicate the level of relevancy of each variable in a Gaussian mixture model (GMM-FS). All the parameters were learned using the EM algorithm. In Zhu et al. [2012], the GMM-FS was revisited and a SVB Bayesian algorithm was used to estimate the model parameters. Li et al. [2009] proposed a localized feature saliency model for a mixture of Gaussians (GMM-LFS), where depending on the mixture component, the set of relevant features could change. The learning process was carried out using a SVB optimization. Guerra et al. [2014] proposed a semi-supervised GMM-LFS to classify partially labeled data. The authors added cluster-dependent feature saliencies to perform the feature selection procedure. Next, Nguyen et al. [2015] developed a mixture model (VB-SMM-LFS), where depending on the cluster, feature saliencies indicated which variables were relevant for the cluster. The clusters were expressed with piecewise-t-Student distributions and the learning of parameters

was performed using SVB methods. Finally, Song et al. [2021] coupled an infinite components piece-wise Gaussian mixture model with localized feature saliencies (iGMM-LFS). The learning phase was performed using a Bayesian method through a Markov chain Monte Carlo algorithm.

3.2.3.3 FSS in HMMs

Concerning HMMs, inspired by the FS-GMMs, Adams et al. [2016] developed a feature saliency HMM (FS-HMM), where a set of feature saliencies were added to the emission probabilities to determine which variables were relevant for the model (as explained in Section 2.3.6). The emission distribution assumed full independence between variables and a maximum a posteriori approach was used to learn the parameter models. Additionally, the model was extended to hidden semi-Markov models, where the sojourn times could be modeled in order to not strictly follow a geometric distribution, as in any traditional HMM. Later, Adams and Beling [2020] proposed an FS-HMM for discrete features in HMMs. It was assumed that the relevant features followed a state-dependent Poisson distribution; whereas, irrelevant features followed a state-independent Poisson distribution. The author provided an EM algorithm to learn a discrete model. In another approach, Zheng et al. [2018] introduced an HMM where the emission probabilities were modeled as mixtures of t-Student distributions (SHMM-LFS). Feature saliencies were added to the model at the component level such that the model was capable of determining, depending on the hidden state and mixture component, which features were noise or relevant. The learning procedure was performed with SVBayesian methods.

3.3 Online modeling for prognosis

In this section, current methodologies concerning the diagnosis (estimate health status) and prognosis (predict remaining useful life) of mechanical components are summarized. The state of the art is divided into three types of methodologies: requires run to failure (RTF) data, requires RTF times, and neither requires RTF data nor RTF times. The last type is more robust and ideal for many real and industrial applications, since the failure data may be non-existing, limited (such as imbalanced data), or expensive in time and money to obtain. This review serves to understand the contributions in Chapter 7, which are also exposed in Puerto-Santana et al. [2022a].

In Table. 3.3 a list of the reviewed literature is presented. The literature is compared depending on:

(**1**.) Does it use HMMs?

(**2**.) Can it be used in online environments?

(**3**.) Does it not require RTF data?

(**4**.) Can it estimate the tool RUL?

(**5.**) Can it update itself when new trends in data appear?

Reference	(**1.**)	(**2.**)	(**3.**)	(**4.**)	(**5.**)
Requires RTF data					
MHMM (Lee et al. [2010])	✓	✓	-	-	✓
Online MoG-HMM (Tobon et al. [2012])	✓	✓	-	✓	-
HMM and NF (Soualhi et al. [2014])	✓	-	-	-	-
HSMM (Cartella F. and H. [2015]	✓)	-	-	✓	-
HMM for TWM (Li and Liu [2019]	✓)	-	-	✓	-
HMM ensemble (Kumar et al. [2019])	✓	✓	-	✓	-
Neo Fuzzy (Soualhi et al. [2013])	-	-	-	-	-
FDFDA (Zhao and Gao [2017])	-	✓	-	-	✓
ANN for TWM (Abdeljaber et al. [2019])	-	-	-	-	-
RNN with HI (Chen et al. [2020])	-	-	-	✓	-
Fault effects (Lin et al. [2021])	-	✓	-	✓	-
LSTM-SVM (Chen et al. [2021])	-	✓	-	✓	-
LSTM with PF (Jiao et al. [2021])	-	✓	-	✓	-
BDNN-RF (Li et al. [2021])	-	✓	-	-	-
Requires RTF times					
Regression (Gebraeel et al. [2009])	-	✓	-	✓	✓
EKM for TWM (Singleton et al. [2015])	-	✓	-	✓	✓
Not RTF times or data are required					
WPD-HMM (Ocak et al. [2007])	✓	✓	✓	-	-
AHMM (Yu [2017])	✓	✓	✓	-	✓
Trigger regression (Li et al. [2015])	-	✓	✓	✓	✓
APCMD (Wu et al. [2019])	-	✓	✓	✓	✓
HSIC (Mohammadi-Ghazi et al. [2020])	-	✓	✓	-	✓
Random Forest (Liu et al. [2021])	-	✓	✓	-	-
Genetic HMMs (Khan and Abuhasel [2021])	✓	✓	✓	-	-
AMBi-GAN (Kong et al. [2021])	-	✓	✓	-	✓
Online As-HMM (Puerto-Santana et al. [2022a])	✓	✓	✓	✓	✓

Table 3.3: State-of-the-art models for prognosis and diagnosis of dynamical systems

3.3.1 Requires run to failure data

Concerning HMMs, the following contributions were found: Lee et al. [2010] proposed an adaptative modified HMM (MHMM) for online tool wear monitoring (TWM). The authors used the forward-backward algorithm and the Hotteling multivariate control chart to detect changes in bearing wear. An online learning algorithm based on the EM algorithm was proposed to update the MHMM. Next, Tobon et al. [2012] proposed an online MoG-HMM for bearings prognosis and RUL prediction. Using the Viterbi algorithm, the distribution of the RUL for each hidden state in RTF data was learned. In the online phase, the learned model and RUL distribution were used to predict the RUL. Later, Soualhi et al. [2014] used HMMs and neuro-fuzzy algorithms (NF) to predict future bearing conditions. First, the data

were clustered to determine different levels of degradation. For each learned cluster, an HMM was learned; when new instances arrived, the most likely HMM was used with a neuro-fuzzy algorithm to predict the future wear of bearings.

Cartella F. and H. [2015] proposed an HSMM for the RUL prediction of bearings. A wrapper algorithm based on the Akaike information criterion was used to determine the best model to predict from a set of models where the time duration distribution, number of components, and number of hidden states change. More recently, Li and Liu [2019] proposed an HMM to predict RUL for TWM. The HMM parameters were computed using a physical ball bearing degradation model, whereas a multilayer perceptron was used to estimate the emission probabilities. The prognosis phase, or RUL prediction, was performed with a modified forward variable. Finally, Kumar et al. [2019] used a MoG-HMMs to determine the RUL of bearings. An initial HMM based on the first observations from all the signals was trained. The log-likelihoods were computed for further observations with an already trained HMM until the log-likelihood surpassed a certain threshold; then, another HMM was trained from the not-well fitted data. RUL was predicted using a polynomial regression function of the log-likelihoods of the learned HMMs.

All the approaches presented here vary their way to predict the degradation and RUL and learn the HMM model. Nevertheless, the need for previous RTF data limits the application of these models and in some cases it is critical for the learning phase as in Tobon et al. [2012], where the distribution of the RUL must be computed for each failure mode and hidden state, making it unfeasible in certain real industrial scenarios. It is also relevant to mention that the prognosis in these methods is usually driven by the Viterbi algorithm and the forward-backward algorithm, with the latter used as well to compute the log-likelihood of data.

Other methodologies not based on HMMs with the hypotheses of RTF data can be found in the state of the art: Soualhi et al. [2013] proposed a long prediction scheme for bearing condition prediction using a model based on fuzzy logic and a single neuron artificial neuronal network (ANN) called neo-fuzzy neuron. Later, Zhao and Gao [2017] proposed an online surveillance process called FDFDA to detect a failure in a dynamical process. A Fisher discriminant analysis was performed to extract relevant components from the data. A ratio of stability factors of faulty data to normal data was computed for each component. Components that went over a certain threshold were eliminated from the analysis and used in an autoregressive regression model to predict the degradation process. Next, Abdeljaber et al. [2019] proposed two 1-dimensional convolution ANNs to determine bearing degradation. One convolutional neural network was used to detect outer ring failures and another to detect inner ring failures. The online testing phase consisted of passing the vibrational signals through both models and classify every instance as healthy or faulty. Chen et al. [2020] used a recurrent neural network (RNN) to predict the RUL of bearings. Time and frequency features were extracted to feed the RNN, and from its output, a linear regression was used to build a health index and predict the bearing RUL. Chen et al. [2021] used a long short time memory network (LSTM) and a support vector regression for RUL prediction of aero-engine data. A risk-averse function was optimized to minimize RUL overestimation.

Lin et al. [2021] proposed a methodology for RUL prediction of turbofan engines. For each feature, a Wiener process was learned and a RUL distribution was estimated using a Bayesian approach considering fault effects. A copula was used to generate a joint RUL distribution. Jiao et al. [2021] used a LSTM network to extract features to determine a normalized health index that predicted the RUL for rollers in hot rolling production. For the RUL estimation, a state space model was used with a particle filtering algorithm for inference and prediction. Whenever a new instance arrived, the state space model could self-update using all the historic data. In spite of that, the LSTM network had to be trained with RTF data. Finally, Li et al. [2021] used a binary deep neural network (BDNN) with random forest (RF) to classify ball bearing faults. The BDNN-RF was designed to be used in edge devices to enable them for online analysis. However, the training phase was performed in a cloud device.

In these non-HMM-based techniques, it was observed that a relevant portion of them used ANN to classify the level of degradation; then, a health index was used to predict the degradation level, and regression was later used to perform the RUL or degradation prediction. Although the ANNs in the previous works were not so computationally demanding, the requirement of previous RTF data can make such methodologies unfeasible in many industrial scenarios.

3.3.2 Requires run to failure times

Sometimes, previous RTF data is not available; however, the records of RTF times are available and can be used for prognosis purposes. However, not many works under these circumstances were found. For instance, Gebraeel et al. [2009] proposed two bearing online prognosis techniques based on exponential and linear regression. The methodology assumed that there was at least prior knowledge of RTF times; from these times, the parameters of a prior Bernstein distribution were estimated using the maximum likelihood method. A threshold from the standard normative was imposed in the magnitude of the ball bearing signals to indicate the start of the failure phase. Later, the residual life distribution was computed, and the RUL was extracted by computing the median of the resultant distribution. Singleton et al. [2015] introduced a bearing online RUL prediction algorithm based on extended Kalman filters (EKF). Whenever a new instance arrived, relevant features were computed and the EKF were updated and used to extrapolate the current signal up to a failure threshold. For determining the threshold, previous RTF times were required.

In these methodologies, the RTF times were used to compute the RUL distribution or indicate the prognosis failure thresholds. However, the prognosis phases were done using regression techniques. It must be recalled that these methodologies can be more flexible than those described in the previous subsection since if RTF data are available, the RTF times are also available; nevertheless, the opposite is not valid.

3.3.3 Does not require RTF data nor RTF times

First, some techniques related to HMMs are summarized. For instance, Ocak et al. [2007] proposed an online algorithm to determine the level of degradation of bearings based on the wavelet packet decomposition (WPD) and HMMs. The methodology extracted its features applying the WPD to the bearing vibrational signals. Then, from the early stages of the bearing life, the signal measures were used to train an HMM. Next, the likelihood from incoming vibrational signals was computed. If the likelihood went below a certain threshold, the bearing behaviour was considered abnormal. Later, Yu [2017] used an online adaptative HMM (AHMM) to determine machine tool wearing. The author used a progressive learning algorithm and a split and merge operation for components in hidden states to update the model whenever a new instance arrived. A Hotteling multivariate control chart was used to detect outliers, and a threshold (set by trial and error) for consecutive out-of-chart instances was imposed to determine when a new hidden state had to be added. Finally, a health index based on the Cauchy-Schwarz correlation between the current HMM and the previous learned HMM was proposed as a flag for degradation.

Second, for non-HMM-based techniques, much more developed work can be found, especially related to regression techniques and novelty detection in data streams. For instance, Li et al. [2015] proposed a regression method for RUL prediction. The authors applied an adaptive approach based on three standard deviation intervals to determine a time trigger to launch an exponential regression model for RUL prediction. Next, Wu et al. [2019] proposed an online prognosis technique for computing ball bearing RUL called APCMD. A principal component Mahalanobis distance algorithm was used to reduce the feature dimensionality and build a health index. The RUL prediction was divided into two phases; the first one built local exponential regressions to represent the local degradation trajectory; in the second one, an empirical Bayesian algorithm was built to predict a global RUL. In Mohammadi-Ghazi et al. [2020], a dependence analysis was used to detect bending and deformations in steel structures. In their model, the authors computed, for each pair of captured features, the normalized Hilbert-Schmidt independence criterion to measure the level of dependency among them. From these measures, a dependency graph was computed. Changes in the health state of the steel structure were deduced from significant changes in the dependency graph.

More recently, Liu et al. [2021] used random forests and data compression to detect abnormal behavior in industrial components. An abnormal bounded score was built from the random forest to determine the level of abnormality of incoming data. In the case of Khan and Abuhasel [2021], the authors used HMMs to detect abnormal signal behavior in several industrial components. The authors used genetic algorithms to learn the HMM parameters from random sub-sequences of normal behavior data. The last offspring generation of HMMs was used to compute the fitness of incoming data. Abnormal behavior was detected by observing changes in the log-likelihood fitness. Kong et al. [2021] proposed an ANN approach to detect anomalies based on generative adversarial networks (GANs)[4] and bi-directional

[4]Generative adversarial networks consist of two ANNs, called discriminator and generator. The idea of the generator ANN is to maximize the classification error of the discriminator ANN, whereas the discriminator

LSTM networks (AMBi-GAN). The model used the residuals of both networks to build a global score to detect anomalies and update the model. As a final detail, since the model was an LSTM, it worked and processed faster than other ANNs e.g., RNNs.

It is worth mentioning that when no-run-to failure data is assumed, fewer works using ANN are observed (at least in our research) and more attention is given to detect and measure changes in distribution or parameters within a data stream (novelty detection and concept drifts). Additionally, in the reviewed articles, regression was the key for prognosis when no RTF data is provided. Also, the signal processing phase plays a vital role in such methodologies, since it is needed to extract the relevant information to detect the above mentioned changes in data distribution.

3.4 Online feature subset selection

In this section, articles related to FSS in streams are reviewed. There are two problems in the literature: the FSS in feature streams and in data streams. The former refers to problems where the number of instances is fixed but the number of features increases over time. The latter has a fixed number of features, but the number of instances increases over time. In both problems, it is desired to determine relevant and non-redundant features to explain the incoming data, and hence its underlying generation mechanism. Although both issues will be reviewed, special attention is put into feature selection in data streams for unsupervised problems. Table 3.4 lists the reviewed articles. They serve as a background to make clear the contributions in Chapter 8 and Chapter 9. In particular, the results from Chapter 8 were published in Puerto-Santana et al. [2022d].

3.4.1 FSS in feature streams

When dealing with feature streams, rough sets were popular among the reviewed articles. For example, Zhou et al. [2019a], Lv et al. [2020] adn Zhou et al. [2019b] used rough sets to select features. In the first case the algorithm OFS-A3M was introduced, aiming at optimizing three quantities: relevance, significance and dependency on the selected variables with respect to the target variable. In the second case the algorithm OFSI was introduced, to minimize the feature redundancy, maximize relevancy and discover feature interactions. In the third case, OFS-density was proposed, where a distance based on density of neighbors was defined and a fuzzy system was used to determine redundancy between selected features.

Other strategies not involving rough sets can be found. In Wang and Luan [2019], the authors generated a set of rules to determine active and possible features to be used. Features highly dependent on the class variable via the correlation are used as a regularization test (COFS). Whenever a new feature arrived, the active and possible set of features was updated applying the regularization test. Zhou et al. [2021] developed an algorithm to select features in supervised feature streams called online group streaming feature selection with feature

ANN learns by minimizing the classification error obtained from the ouput generated by the generator ANN.

Topic	Name	Reference
Feature streams:	OFS-A3M	Zhou et al. [2019a]
	OFSI	Lv et al. [2020]
	OFS-density	Zhou et al. [2019b]
	COFS	Wang and Luan [2019]
	OGSFS-FI	Zhou et al. [2021]
	OCFSSF	You et al. [2021]
	UFSSF	Almusallam et al. [2021]
Data streams:	**Supervised**	
	DXMiner	Masud et al. [2010]
	DXMiner-FI	Wang and Shen [2016]
	DISCUSS	Barddal et al. [2019]
	HEFT-Stream	Nguyen et al. [2012]
	OFS-perceptron	Hoi et al. [2012]
	RAC	Hoeltgebaum et al. [2021]
	Unsupervised	
	OLFS-DMM	Fan and Bouguila [2013]
	sketching matrix	Huang et al. [2015]
	MV-FS	Shao et al. [2016]
	FS-K-means	Wang et al. [2018]
	DFM-MCFS	Fahy and Yang [2019b]
	FSMCP	Ma et al. [2020]
	Online FS-HMM	Puerto-Santana et al. [2022d]
	Online LFS-AsHMM	This thesis

Table 3.4: Table of reviewed articles for online FSS in data streams and feature streams

interaction (OGSFS-FI). In the article, mutual information between variables was used to compute the information gain when interaction between features are considered. An online algorithm was provided to determine the strong relevancy, weak relevancy and redundancy of incoming group of features, in order to update the set of selected features. You et al. [2021] introduced the online causal feature selection with streaming features (OCFSSF). The algorithm searched for the best Markov blanket for a class variable. The methodology adapted parents, children and spouses of incoming features in the Markov blanket of the class variable. The changes depended on scores such as strong relevancy, weak relevancy and redundancy. Finally, Almusallam et al. [2021] proposed an unsupervised feature selection with feature stream (UFSSF). The methodology assumed that the number of relevant features was fixed. Whenever a new feature arrived, a similarity measure was used to determine the most related feature to the one that arrived. The new feature replaced the most similar feature.

Observe that the majority of the reviewed methodologies were designed for supervised problems. Also, univariate filter feature selection strategies were mostly used.

3.4.2 FSS in data streams

3.4.2.1 Supervised

The earliest work found tackling the problem of feature selection in data streams is Masud et al. [2010], where the DXMiner was introduced as a model to face infinite data stream processes with concept drift, feature drift and concept evolution. DXMiner used a finite ensemble of semi-supervised K-nearest neighbor (K-NN) classifiers that were updated whenever a new chunk of data arrived. The feature selection algorithm was carried out with a univariate filter feature selection algorithm over the incoming data. Later, Wang and Shen [2016] changed the feature selection algorithm and used instead the multi-cluster feature selection model, where interactions between features were taken into account for a better feature selection procedure (DXMiner-FI). Another filter methodology was found in Barddal et al. [2019]. The authors proposed the dynamic symmetrical uncertainty selection for streams (DISCUSS). This methodology used a filter multivariate feature selection based also on the symmetrical uncertainty score to select features. Whenever a new instance arrived, the score of each feature was updated.

Some authors have tried to apply boosting or ANNs to learn feature relevancy. Regarding boosting, Nguyen et al. [2012] proposed the heterogeneous ensemble with feature drift for data stream algorithm (HEFT-Stream). The authors generated an ensemble of classifiers of different types where each classifier had its own feature space which was generated or updated when a feature drift was detected. The feature selection was conducted using a multivariate filter methodology, where the features that optimized the symmetrical uncertainty were selected. In the case of neural networks, the embedded methodology of Hoi et al. [2012] was found. The authors proposed an online embedded feature selection algorithm based on the perceptron model (OFS-perceptron). A truncation on the weights of the network was set to select the most relevant features. The algorithm learned the weights using an online descent gradient algorithm. Finally, another embedded methodology was found: Hoeltgebaum et al. [2021] proposed the real-time adaptative component (RAC) based on streaming Lasso regression for supervised data. In this case, the relevant features were determined with the coefficients of the Lasso regression. Also, an online methodology was proposed to determine dynamically the penalization parameter to select features and the forgetting factors.

3.4.2.2 Unsupervised

Since this is the main topic of this thesis, the reviewed articles are grouped depending on whether a filter, embedded or wrapper methodology was used to select features. In the case of embedded methodologies, Fan and Bouguila [2013] proposed an online Dirichlet mixture model with localized feature saliencies (OLFS-DMM). The authors added a latent Bernoulli variable to distinguish between relevant and irrelevant features. The learning process was carried out using an online variational Bayesian method, which updated the model whenever a new instance arrived. Other embedded methodologies used penalized regression strategies: Huang et al. [2015] proposed an unsupervised feature selection algorithm based on matrix

sketching. The algorithm used the singular value decomposition on a low rank representation of the full dataset (matrix sketching) and a ranking of components to create a penalized regression problem. From the regression, the obtained weights were used as a relevancy score to filter the features. In Shao et al. [2016], it was assumed that several datasets could arrive simultaneously or there are multiple views and the goal was to determine for each dataset, their relevant features (MV-FS). A non-negative matrix factorization was carried out for each dataset coming from each view using a penalized quadratic optimization problem. The output of the optimization was a feature relevancy vector. Finally, Wang et al. [2018] proposed an online K-means algorithm with a feature selection methodology (FS-K-Means). The authors formulated the K-means optimization problem in terms of indicator matrices, cluster centroid matrices and projection matrices. The projection matrix estimations from the learning phase were used to determine the feature relevancy.

With respect to filter methodologies, Fahy and Yang [2019b] proposed a dynamic FSS algorithm that could be used together with any model-based clustering (DFM-MCFS). Their idea was to perform a cluster univariate feature selection for clustering once a buffer of data was filled. The selected features were used to update a relevancy vector, which indicated the pertinent features. The features that overpassed a threshold were used in the cluster model. Finally, Ma et al. [2020] created an online feature selection algorithm based on multi-cluster structure preservation (FSMCP). First, the methodology computed the membership probabilities of arriving instances to current clusters. Then, a penalized Kullback-Leibler was minimized in order to obtain a set of weights that were used as feature relevancy.

With respect to wrapper methodologies, no articles were found for the case of unsupervised data in data stream environments. A possible reason behind this is that wrapper methodologies require high computational effort and time which is not feasible in the case of online problems, where the response times from algorithms must be fast.

3.5 Kernel density estimation

In Chapter 10, a new family of HMMs, where the emission probabilities are expressed with KDEs and context-specific Bayesian networks is introduced. Thus, the relevant bibliography related to KDEs with Bayesian networks and HMMs is firstly reviewed. We will note that, no previous work merges KDEs, Bayesian networks and HMMs. Therefore, the proposed model, KDE-AsHMM is capable of describing non-Gaussian distributions, taking into consideration data interactions. In Table. 3.5

3.5.1 Bayesian networks and KDE

Hofmann and Tresp [1995] proposed a heuristic to learn Bayesian networks with kernel conditional estimation (KCE). The networks were learned using leaving-one-out scores penalized by the number of arcs in the model. In this model, each feature had its own bandwidth parameter to be estimated. Pérez et al. [2009] introduced kernel-based Bayesian networks for

classification, where the bandwidths were computed using rule of thumb formulas, see Silverman [1986]. Through statistical testing they found that the tree augmented naive Bayes obtained the minimum prediction error for supervised datasets. Finally, Atienza et al. [2022] proposed a semi-parametric Bayesian networks where the dependencies between variables were defined combining a linear Gaussian Bayesian network fashion and KCE. The authors proposed a graph search algorithm based on the tabu meta-heuristic, see Glover [1986]; in the graph search, each node could change its dependency model (linear Gaussian or kernel-based) depending on how that improved the cross-validated log-likelihood.

Topic	**Summary**	**Cite**
Bayesian networks	Introduced KCE in Bayesian Networks	Hofmann and Tresp [1995]
	KCE in Bayesian classifiers	Pérez et al. [2009]
	Context-specific LGBN in HMMs	Puerto-Santana et al. [2022b]
	LGBN and KCE for Bayesian networks structure	Atienza et al. [2022]
HMMs	Markov process with KDE	Rajarshi [1990]
	Input-output KDE in HMMs	Wang et al. [2003]
	AR values as kernel centres with KDE emission probabilities	Xu et al. [2005]
	KDE-HMMs treated as MoG-HMMs	Piccardi and Perez [2007]
	KDE in MoGs/HMMs with ANNs	Do et al. [2014]
	KDE in Markov random fields with Gibbs sampling	Qiao and Xi [2017]
	AR-HMM conditional emission distribution defined by KDE	Henter et al. [2018]
	KDE in transitions in HSMMs	Luati and Novelli [2021]
	KDE in the covariance in HMMs with Gaussian processes	Jung and Park [2022]
	HMMs with KDE emissions with context-specific Bayesian netwroks	This thesis

Table 3.5: Reviewed articles about Bayesian networks, KDEs and HMMs

3.5.2 HMMs adn KDEs

Rajarshi [1990] proposed a bootstrap method based on KDE to estimate stationary Markov processes. Although no hidden variable was used in this first work, these ideas were generalized to be adapted in HMMs, as it will be seen below. Wang et al. [2003] proposed an HMM with input-output observations. The model assumed that in each hidden state, the output variables would depend on the input variables in a manner described by a Bayesian networks, and the conditional probabilities were estimated using KCE. The authors proposed an EM algorithm with a Monte Carlo sampling phase in the M-step to reduce the number of instances in the kernel and accelerate the training phase. Xu et al. [2005] proposed a kernel-based HMM, where the emission probabilities were a KDE model whose kernel took as arguments the current and last observations. The model learning procedure was modified

to maximize the accuracy in a supervised problem. Another approach to join HMMs and KDEs was proposed in Piccardi and Perez [2007], where an HMM with kernel-based emission probability estimation was proposed, but, in this case, a pseudo-likelihood function was used to run the EM algorithm. For the multivariate kernel, the bandwidths were defined with a matrix in order to take into consideration feature interactions. Do et al. [2014] proposed a kernel-based density HMM for classification in speech recognition. The emission probabilities were based on KDEs; however, in this case, a global bandwidth parameter was used. Next, an ANN was used to re-estimate the aposteriori probabilities of the cluster/class variable to improve the accuracy of the speech recognition.

In more recent years, further work related to HMMs and KDEs can be found. Qiao and Xi [2017] proposed a kernel-based hidden Markov random field model, where the emission probabilities were modeled as KDEs; but the latent probabilities were modeled with a Gibbs distribution. An EM algorithm was proposed to determine the model parameters, in particular, the bandwidth of the kernels depended on the hidden state. Henter et al. [2018] proposed an AR-HMM where a KDE was used to define the conditional next-step emission distributions. An extended EM algorithm was applied to learn the model parameters; also, the bandwidths and kernel centers were dependent on each hidden state. This was subsequently extended to kernel conditional density estimation by De Gooijer et al. [2022]. In Luati and Novelli [2021], the KDEs were used to estimate the sojourn times distributions for explicit state duration in hidden semi-Markov models (HSMM). Finally, Jung and Park [2022] proposed an HMM with Gaussian processes as emission probabilities. For the covariance function, a spectral mixture kernel was applied, and the model parameters were learned using variational Bayesian methods.

Part II

CONTRIBUTIONS TO HMMs

Chapter 4

Asymmetric linear Gaussian HMMs

The model proposed in this chapter is an extension of that presented in Section 2.3.5. As-HMMs were designed to work with discrete variables; therefore, if continuous variables are being used, a discretization step must be done. Clearly, such step can add undesirable noise to the model parameters. With the proposed model in this chapter, continuous variables can be directly used to learn an As-HMM. This chapter is a summary of Puerto-Santana et al. [2018].

The proposal uses LGBNs to model state-specific Bayesian networks; therefore, it is assumed that the observable variables behave as a linear Gaussian distribution given the hidden state. Thus, the model, as previously reviewed in Section 3.1.2, is called asymmetric linear Gaussian HMM (AsLG-HMM). The model is only compared with MoG-HMMs. However, in the next chapters, extensions of AsLG-HMMs will be compared with other HMMs. The chapter organization is as follows: In Section 4.1 we define the new model. In Section 4.2, we obtain the parameters updating formulas based on the EM algorithm. Section 4.3 describes the data and experimental results used for model validation. Section 4.4 provides the corresponding conclusions and comments.

4.1 Model proposal

An AsLG-HMM over the continuous observable variables $\boldsymbol{X}^t = (X_1^t, ..., X_M^t)$, and the hidden discrete variable Q^t, $t = 0, ..., T$, is an As-HMM $\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, with the property that for each $q \in R(Q^t)$ a state-specific linear Gaussian Bayesian network $(\boldsymbol{\theta}_q, \mathcal{B}_q)$ is associated. If the k_{qm} parents of the variable X_m^t for the state q are $\mathbf{Pa}_q(X_m^t) = (U_{qm1}^t, ..., U_{qmk_{qm}}^t)$, $m = 1, ..., M$, with $U_{qml}^t \in \text{Set}(\boldsymbol{X}^t)$, $l = 1, ..., k_{qm}$, then the emission probabilities $\mathbf{B} = [b_i(\boldsymbol{x}^t)]_{i=1}^N$ have the analytic form:

$$b_i(\boldsymbol{x}^t) = \prod_{m=1}^{M} \mathcal{N}\big(x_m^t | \beta_{im0} + \beta_{im1} u_{im1}^t + \cdots + \beta_{im,k_{im}} u_{imk_{im}}^t, \sigma_{im}^2\big), \tag{4.1}$$

if $\boldsymbol{\beta}_{im} = (\beta_{im0}, ..., \beta_{imk_{im}})^\top$ and $\boldsymbol{u}^t_{im} = (1, u^t_{im1}, ..., u^t_{imk_{im}})$; then, the previous equation can be written in a more compact way:

$$b_i(\boldsymbol{x}^t) = \prod_{m=1}^{M} \mathcal{N}\big(x^t_m | \boldsymbol{u}^t_{im}\boldsymbol{\beta}_{im}, \sigma^2_{im}\big), \tag{4.2}$$

It is assumed the same hypotheses about transition probabilities between hidden states and observable variables as in traditional HMMs. Therefore, no modification must be done to the forward-backward algorithm or the Viterbi algorithm.

4.2 Learning AsLG-HMMs

4.2.1 Learning the parameters

Now that it is known how to represent the emission probabilities $\mathbf{B}$ for the case of AsLG-HMMs, the corresponding EM parameter updating formulas are deduced. Assume the prior $\boldsymbol{\lambda}^{(s)}$ is known and the E step is executed as in Section 2.3.2. Therefore $\gamma^t(i)$ and $\xi^t(i,j)$, $i,j = 1, ..., N$, $t = 0, 1, ..., T$ are already calculated. For our model the auxiliary function $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ for the M step is:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \gamma^0(i)\ln(\pi_i) + \sum_{t=0}^{T-1}\sum_{j=1}^{N}\sum_{i=1}^{N} \xi^t(i,j)\ln(a_{ij}) + \\ \sum_{t=0}^{T}\sum_{i=1}^{N}\sum_{m=1}^{M} \gamma^t(i)\ln(\mathcal{N}\big(x^t_m|\boldsymbol{u}^t_{im}\boldsymbol{\beta}_{im}, \sigma^2_{im}\big)),\end{aligned} \tag{4.3}$$

the Lagrangian function for this model is:

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + \mu_0(1 - \sum_{i=1}^{N}\pi_i) + \sum_{i=1}^{N}\mu_i(1 - \sum_{j=1}^{N} a_{ij}). \tag{4.4}$$

The updating formulas of the parameters β_{im0} are found. Hence Eq. (4.4) is derived with respect to β_{im0} and equated to zero. Hence :

$$\begin{aligned}\frac{\partial\mathcal{L}(\boldsymbol{\lambda})}{\partial\beta_{im0}} = &\sum_{t=0}^{T}\gamma^t(i)\frac{\partial}{\partial\beta_{im0}}\ln(\mathcal{N}\big(x^t_m|\boldsymbol{u}^t_{im}\boldsymbol{\beta}_{im}, \sigma^2_{im}\big)) \\ &\sum_{t=0}^{T} -2\frac{\gamma^t(i)}{\sigma^2_{im}}(\boldsymbol{u}^t_{im}\boldsymbol{\beta}_{im} - x^t_m) = 0, \\ &\sum_{t=0}^{T}\gamma^t(i)x^t_m = \sum_{t=0}^{T}\gamma^t(i)\boldsymbol{u}^t_{im}\boldsymbol{\beta}_{im}.\end{aligned} \tag{4.5}$$

Now, if Eq. (4.4) is derived with respect to the coefficients $\boldsymbol{\beta}_{imk}$ with $k = 1, ..., k_{im}$ as in

Eq. (4.5), the following equations are obtained:

$$\begin{cases} \sum_{t=0}^{T} \gamma^t(i) x_m^t u_{im1}^t & = \sum_{t=0}^{T} \gamma^t(i) u_{im1}^t \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} \\ & \vdots \\ \sum_{t=0}^{T} \gamma^t(i) x_m^t u_{imk_{im}}^t & = \sum_{t=0}^{T} \gamma^t(i) u_{imk_{im}}^t \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}. \end{cases} \tag{4.6}$$

Eq. (4.5) and Eq. (4.6) form a linear system of $k_{im}+1$ unknowns with $k_{im}+1$ equations which can be solved with methods such as Gauss-Jordan, LU decomposition or Cholesky decomposition. Nevertheless, a closer look at the previous equation can provide a closed formula for the parameter $\boldsymbol{\beta}_{im}$ updating formula. If $\boldsymbol{\Gamma}_i^{0:T} := \text{Matrix}([\gamma^0(i), ..., \gamma^T(i)])$, then the updating formula of $\boldsymbol{\beta}_{im}$ from the linear equation system from Eq. (4.5) and Eq. (4.6) can be written as the LSE of a weighted regression:

$$\begin{aligned} (\boldsymbol{u}_{im}^{0:T})^\top \boldsymbol{\Gamma}_i^{0:T} \boldsymbol{x}_m^{0:T} &= (\boldsymbol{u}_{im}^{0:T})^\top \boldsymbol{\Gamma}_i^{0:T} \boldsymbol{u}_{im}^{0:T} \boldsymbol{\beta}_{im} \\ \boldsymbol{\beta}_{im}^{(s+1)} &= \left((\boldsymbol{u}_{im}^{0:T})^\top \boldsymbol{\Gamma}_i^{0:T} \boldsymbol{u}_{im}^{0:T} \right)^{-1} (\boldsymbol{u}_{im}^{0:T})^\top \boldsymbol{\Gamma}_i^{0:T} \boldsymbol{x}_m^{0:T} \end{aligned} \tag{4.7}$$

To obtain the updating formula of σ_{im}^2, Eq. (4.4) is derived with respect to σ_{im}^2 and equate to zero:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \sigma_{im}^2} &= \sum_{t=0}^{T} \gamma^t(i) \frac{\partial}{\partial \sigma_{im}^2} \ln(\mathcal{N}(x_m^t | (\boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)}, \sigma_{im}^2)) \\ 0 &= \sum_{t=0}^{T} \gamma^t(i) \left(\frac{(x_m^t - \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)})^2}{\sigma_{im}^4} - \frac{1}{\sigma_{im}^2} \right) \\ (\sigma_{im}^2)^{(s+1)} &= \frac{\sum_{t=0}^{T} \gamma^t(i) \left(x_m^t - \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} \right)^2}{\sum_{t=0}^{T} \gamma^t(i)}. \end{aligned} \tag{4.8}$$

The complexity of computing the $\boldsymbol{\beta} = \{\boldsymbol{\beta}_{im}\}_{i=1,m=1}^{N,M}$ coefficients is discussed: assume that the factorization for each state is the most complex i.e., every variable depends on the others. This implies that $|\boldsymbol{\beta}| = N(1+2+3+\cdots+M) = \frac{NM(M+1)}{2}$. It is known that the complexity of solving a linear system of k variables is usually $O(k^3)$ (using for example the Gauss-Jordan algorithm). Hence, for the worst case scenario the complexity of determining the coefficients for a single state is $O(\sum_{m=1}^{M} m^3) = O\left(\frac{M^2(M+1)^2}{4}\right)$. Therefore to compute the coefficients for every state, the complexity is $O(NM^4)$. On the other hand, for the simplest factorization i.e., every variable is independent of the others given the state, the complexity of determining the coefficients is $O(NM)$, since M divisions for N states must be done.

4.2.2 Learning the structure

The structure learning procedure in the framework of AsLG-HMM is done as in the Section 2.2.2. Recall that in the SEM algorithm, it must be found a model $\mathcal{B}^{(l+1)}$ that maximizes

$\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)})$. For the case of AsLG-HMMs, $\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)})$ is:

$$\begin{aligned}\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)}) = \sum_{i=1}^{N} \gamma^0(i) \ln(\pi_i) + \sum_{t=0}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi^t(i,j) \ln(a_{ij}) + \\ + \sum_{t=0}^{T} \sum_{i=1}^{N} \sum_{m=1}^{M} \gamma^t(i) \ln(\mathcal{N}\big(x_m^t | \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}, \sigma_{im}^2\big)) - 0.5\#(\boldsymbol{\lambda}) \ln(T+1).\end{aligned} \tag{4.9}$$

where $\mathcal{B}$ for AsLG-HMMs is the set $\mathcal{B} = \bigcup_{i=1}^{N} \{\mathcal{B}_i\}$, with $\mathcal{B}_i$ representing the state-specific Bayesian network graph of state i. It can be noticed that each $\mathcal{B}_i$ is related with the expression $\gamma^t(i) \ln(\mathcal{N}\big(x_m^t | \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}, \sigma_{im}^2\big)) - 0.5\#(\boldsymbol{\lambda}) \ln(T)$. Therefore, due to the linearity of the problem, such expression can be used to find $\mathcal{B}^{(l+1)}$ that maximizes Eq. (4.9). Then, the next scores are introduced:

$$\begin{aligned}\text{score}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)}) &= \sum_{i=1}^{N} \text{score}_i(\mathcal{B}_i, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)}) \\ \text{score}_i(\mathcal{B}_i, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)}) &= \sum_{t=0}^{T} \sum_{m=1}^{M} \gamma^t(i) \ln(\mathcal{N}\big(x_m^t | \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}, \sigma_{im}^2\big)) - 0.5\#(\mathcal{B}_i) \ln(T+1).\end{aligned} \tag{4.10}$$

As consequence, instead of using $\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)})$ in the SEM algorithm, it can be used $\text{score}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(l)}, \boldsymbol{\lambda}^{(l,*)})$ to search for the best model structure. For this case study, the optimization process is done with simulated annealing Kirkpatrick et al. [1983].

4.3 Experimental setup

4.3.1 Dataset characterization

The data used in this chapter and incoming chapters, come from ball bearing vibrational information, see Qiu et al. [2006]. The data is filtered using the spectral kurtosis algorithms and envelope techniques explained in Section 2.5. The ball bearing fundamental frequencies and their harmonics are extracted: BPFO, BPFI, BSF and FTF. The mechanical set-up is shown in Fig. 4.1.

In real life applications, ball bearings are fundamental components inside tool machines. It is desirable to monitor their health state. However, the health state is a hidden variable; hence, HMMs can be applied to estimate it. This first proposed model, as a first step, will be compared to a more common model such as the MoG-HMM. A MoG-HMM can be seen as a symmetric model, because its emission probabilities, depending on the modeler decision, in their covariance matrices, must be diagonal or full for all the hidden states. In the former, all the variables are independent, in the later, all the variables can depend on the others, and the number of parameters can increase significantly. If AsLG-HMMs are used, it can be discovered which frequencies affect directly others. Or in other words, it can be inferred which components of the ball bearing are driving its mechanical modes and which others

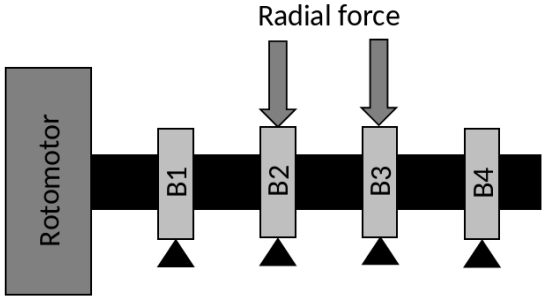


Figure 4.1: Graphical representation of the mechanical set-up. A rotomotor spins a shaft at a rotational speed of 2000RPM coupled with four Rexnord ZA-2115 double row bearings with labels B1, B2, B3 and B4. A constant radial load of 2721.554kg is applied to bearings 2 and 3. Vibrational data is recorded until one of the bearings fails. A signal record of 0.1s is taken every twenty minutes. The sampling rate is 20kHz.

are responding to these behaviors. Three signals datasets are available from this mechanical set-up. Table. 4.1 shows a description of each signal.

Signal	Number of records	Defects found
S1	2156	bearing 3 BPFI and bearing 4 BSF
S2	984	bearing 1 BPFO
S3	4448	bearing 3 BPFO

Table 4.1: Information about the dataset and a description of the failures found in each signal record. Each signal is divided in four signals, one for each bearing. As notation,Su Bv is the information from the u signal dataset for the bearing Bv, $u = 1, 2, 3$ and $v = 1, 2, 3, 4$. Su Bv has information of the fundamental frequencies and harmonics of Bv. Also, it is known that S1 B3, S1 B4, S2 B1, S3 B3 are run to failure signals.

4.3.2 Experimental setup

The experiment consists of learning an AsLG-HMM and MoG-HMM for each signal and ball bearing, and using all the signals for testing. Four hidden states are assumed for all the models. And, models using 0 and 3 harmonics are learned. During the estimations, it was found that the small amplitudes of the frequencies could generate numerical errors and get probabilities larger than 1 or equal to zero; therefore, during the learning process, every variable was scaled in the interval $[0, 10]$, the scaling constant are used to scale the testing variables. In total, 48 models are learned for this experimental set-up (24 from AsLG-HMM and 24 from MoG-HMM).

In order to see the evolution of states of the failing bearings (S1 B3, S1 B4, S2 B1 and S3 B3), the Viterbi algorithm is used. An ideal sequence of states is one where the healthy, fair and failure states can be identified. However, the predicted states must be interpreted from the model parameters to know what they represent, since they are not observable but hidden. For that reason in each model the faulty and the healthy states must be deduced.

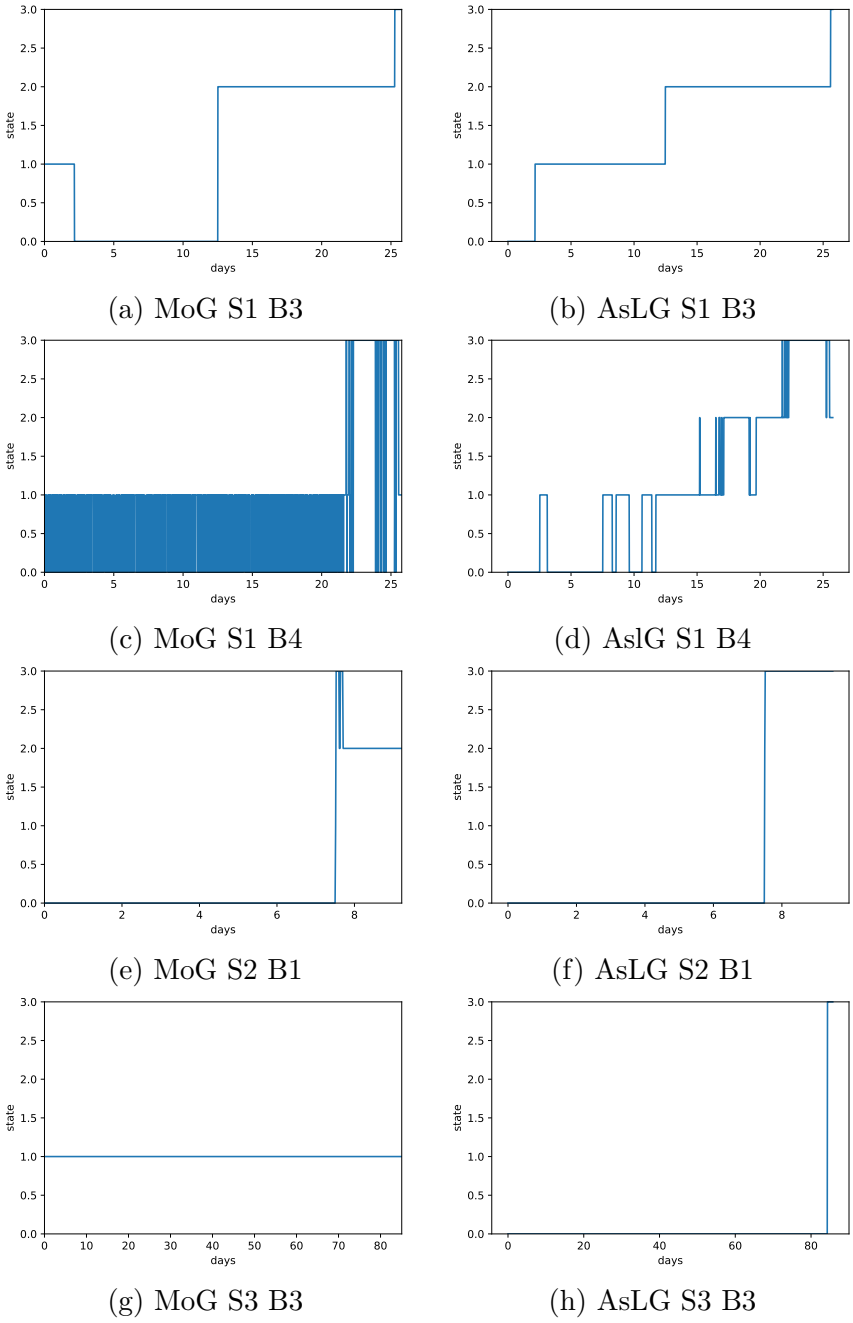
4.3.3 Results

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	92	-13751.94	-14102.85	44	-12174.38	**-12324.90**
S1	B2	92	-13413.10	-13764.01	44	-11484.06	**-11634.58**
S1	B3	92	-9108.65	-9459.56	44	-7202.71	**-7353.23**
S1	B4	92	-8026.09	-8377.00	44	-6446.02	**-6596.55**
S2	B1	92	-165936.05	-166248.14	44	-23150.37	**-23280.93**
S2	B2	92	-9234.61	-9546.69	44	-6437.78	**-6568.34**
S2	B3	92	-6851.84	-7163.93	44	-3936.90	**-4067.46**
S2	B4	92	-3281.82	-3593.91	44	-1908.85	**-2039.41**
S3	B1	92	-41907.36	**-42309.22**	44	-43653.76	-43828.32
S3	B2	92	-51109.17	-51511.04	44	-48086.59	**-48261.15**
S3	B3	92	-102036.99	-102438.86	44	-48625.15	**-48799.71**
S3	B4	92	-24644.64	-25046.51	44	-19073.07	**-19247.64**

Table 4.2: Likelihood and BIC results when S1 is used as training set with 0 harmonics.

Table. 4.2 shows the results obtained from MoG-HMMs and AsLG-HMMs, when signal S1 is used for learning the model parameters and zero harmonics were considered. Observe that the BIC score was always better for the case of AsLG-HMM except for one case. Also, the number of parameters required by AsLG-HMM for all the cases is lower than in the case of MoG-HMM. Regarding the sequences of hidden states for AsLG-HMMs, the are pictured in Fig. 4.2. In literal (b), the sequence for AsLG-HMM for S1 B3 showed a clear evolution of states, where state 0 is the initial or healthy state and state 3 is the faulty state. In (d) for S1 B4, a similar evolution of states up to failure was observed. In (f) for S2 B1, it was no longer seen a step by step evolution of states but rather a shoot at the last days of life of the ball bearing from state 0 to state 3. And in (h) for S3 B3, something similar happened as in (f) . On the other hand, the sequences estimated by the MoG-HMM provided similar or worse results. In (a) for S1 B3, the evolution of hidden states was clear, where 1 is the healthy state, 0 and 2 intermediate states and 3 was the failure state. In (c) for S1 B4, the model did not distinguish between state 1 and 0 as initial or healthy states which is undesirable, also a shoot to state 3 occurred in the last days of life of the bearing, which implies that state 3 was the faulty one. In (e) for S2 B1, the initial state was 0 and failure state was 2 since in its last days, the sequence placed in this state. In (g) for S3 B3, the model said that no evolution occurred on the ball bearing, which is not true.

Table. 4.3 shows the results obtained from the MoG-HMM and AsLG-HMM when signal S1 was used for learning the model parameters and three harmonics were considered. In this case, all the BIC scores were better for the AsLG-HMMs. With respect to the number of parameters, note that it was a constant high value (1100 parameters) for MoG-HMM for all the ball bearings (since it is a symmetric model), and in the case of AsLG-HMM, such number changed depending on the ball bearing, from 133 parameters to 560 (asymmetric behavior). Regarding the Viterbi paths, Fig. 4.3 shows the resulting health state sequences for both AsLG-HMMs and MoG-HMMs. In the case of the results obtained from MoG-HMMs, in (a), (c), (e) and (g) the health state estimations are shown. It is observed that for most of the



(a) MoG S1 B3 (b) AsLG S1 B3

(c) MoG S1 B4 (d) AslG S1 B4

(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.2: Sequences of states determined by the parameters learned from S1 with 0 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

cases, but (g), the health states evolution was not clear and a noisy behavior was present in the estimations. In the case of AsLG-HMMs, in (b), (d), (f) and (h), the Viterbi paths are pictured. For (b) S1 B3 estimation and (d) S1 B4 , a noisy evolution of hidden states

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	1100	-54002.23	-58197.91	560	-48149.07	**-50196.19**
S1	B2	1100	-50993.39	-55189.07	560	-43337.75	**-45384.87**
S1	B3	1100	-37288.72	-41484.40	137	-32174.89	**-32630.22**
S1	B4	1100	-23903.19	-28098.87	133	-23468.56	**-23908.84**
S2	B1	1100	-291107.10	-294838.55	560	-48319.57	**-50095.18**
S2	B2	1100	-37168.59	-40900.04	560	-23117.11	**-24892.72**
S2	B3	1100	-25677.86	-29409.31	137	-16930.24	**-17325.18**
S2	B4	1100	-11094.17	-14825.62	133	-7751.82	**-8133.70**
S3	B1	1100	-161721.44	-166526.33	560	-162654.42	**-165028.46**
S3	B2	1100	-202135.82	-206940.72	560	-178040.19	**-180414.23**
S3	B3	1100	-349800.92	-354605.81	137	-270698.52	**-271226.57**
S3	B4	1100	-76990.41	-81795.30	133	-74495.96	**-75006.56**

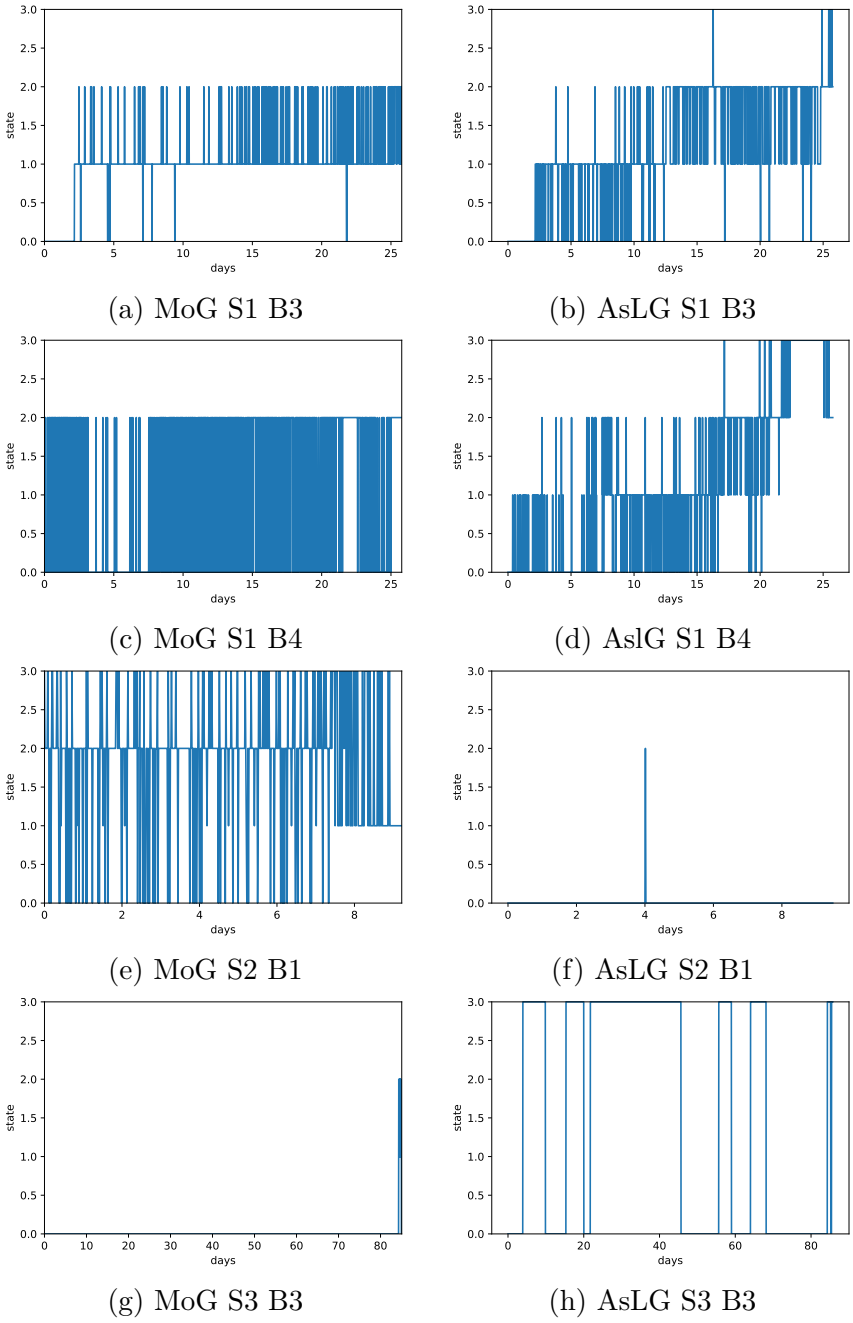
Table 4.3: Likelihood and BIC results when S1 is used as training set with 3 harmonics.

were observed, where 0 was the healthy state and 3 was the faulty state; for the case of S2 B1 (f) and S3 B3 (h) the estimations were not clear and no conclusions could be drawn. In comparison, when no harmonics were used, S1 with AsLG-HMMs could provide an estimation of the time were the ball bearing failed in S2 and S3 (not training signals).

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	92	-25320.08	-25671.00	39	-14815.86	**-14947.57**
S1	B2	92	-26790.53	-27141.44	44	-16107.71	**-16258.23**
S1	B3	92	-30186.46	-30537.37	44	-20666.38	**-20816.9**
S1	B4	92	-68835.30	-69186.21	44	-48431.78	**-48582.3**
S2	B1	92	-3549.33	-3861.41	39	-1677.26	**-1791.5**
S2	B2	92	-5949.62	-6261.71	44	-3483.04	**-3613.60**
S2	B3	92	-6255.21	-6567.29	44	-4875.81	**-5006.37**
S2	B4	92	-4587.89	-4899.97	44	-3144.30	**-3274.86**
S3	B1	92	-18229.13	-18630.99	39	-12740.61	**-12893.35**
S3	B2	92	-58430.15	-58832.02	44	-42388.86	**-42563.42**
S3	B3	92	-188725.90	**-189127.76**	44	nan	nan
S3	B4	92	-37769.36	-38171.22	44	-28058.40	**-28232.96**

Table 4.4: Likelihood and BIC results when S2 is used as training set with 0 harmonics.

Table. 4.4 shows the results obtained from the MoG-HMM and AsLG-HMM when signal S2 was used for learning the model parameters and zero harmonics were considered. In this case, all the BIC scores were better for AsLG-HMMs; however, a numerical error raised when the S3 B3 was tested. This was caused because the probability of the observations at the end of the bearings life were near to zero for the distributions of hidden states. Regarding number of parameters, it was observed that AsLG-HMMs used less parameters than MoG-HMMs and could change depending on the ball bearing. Fig. 4.4 shows the health estimations obtained for all the models. The Viterbi paths corresponding to AsLG-HMM can be seen in (b), (d), (f) and (h). For the ball bearing S1 B3 (b) the sequences could not provide any information; meanwhile for the ball bearing S1 B4 (d), an evolution of hidden states was observed, where 0 was a healthy sate and 3 was a failure state. For S2 B1 (f), the evolution of hidden states says that 1 was a healthy state, 0 and 2 were two intermediate states and 3 was a failure



(a) MoG S1 B3 (b) AsLG S1 B3

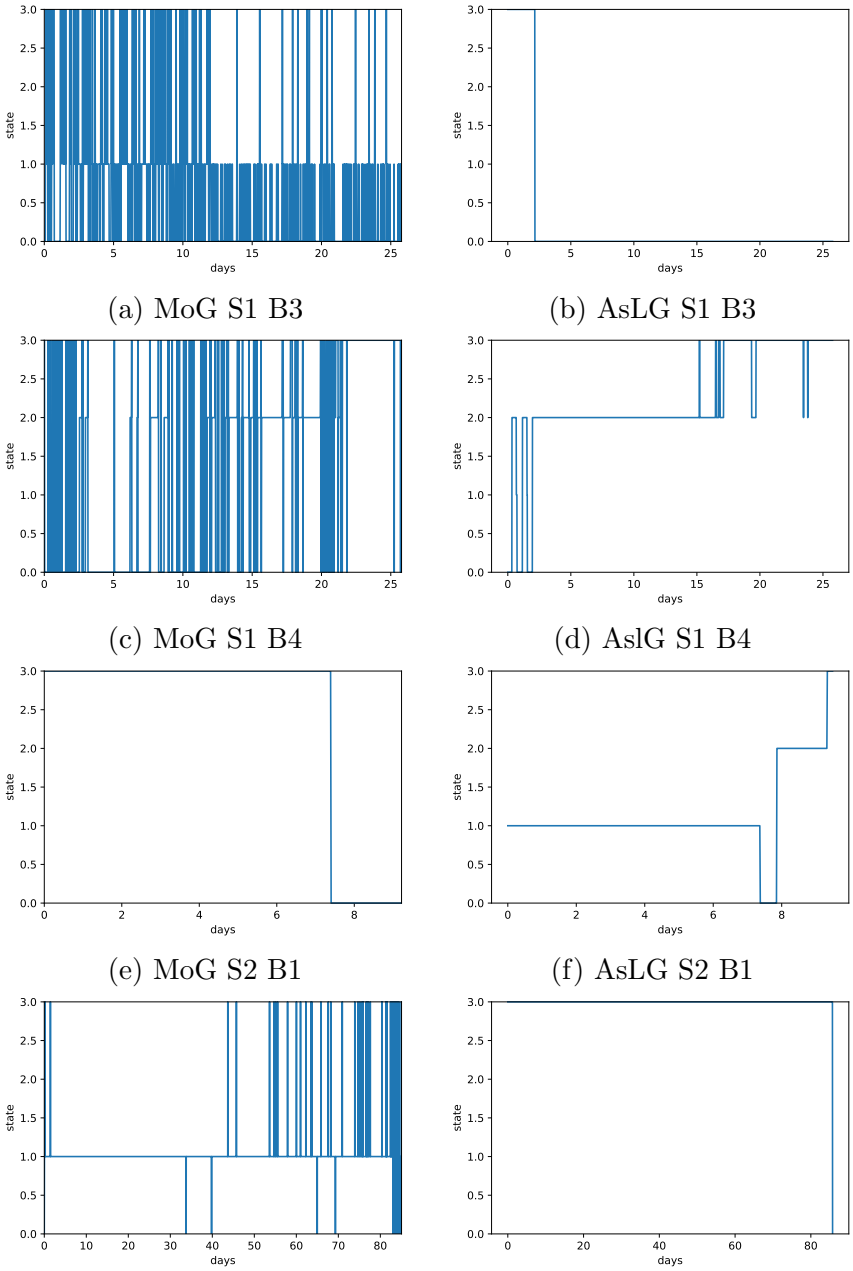
(c) MoG S1 B4 (d) AslG S1 B4

(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.3: Sequences of states determined by the parameters learned from S1 with 3 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

state. For S3 B3 (h), the model estimated a healthy state for most of the ball bearing life to later run to failure state. Regarding the paths obtained by MoG-HMMs, they are drawn in (a), (c), (e) and (g). For most of the ball bearings, but S2 B1 (c), noisy and non interpretable



(a) MoG S1 B3 (b) AsLG S1 B3

(c) MoG S1 B4 (d) AslG S1 B4

(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.4: Sequences of states determined by the parameters learned from S2 with 0 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

segmentations were obtained. For S2 B1(c), it can be said that 3 was a healthy state and 0 was failure state and no intermediate states were present.

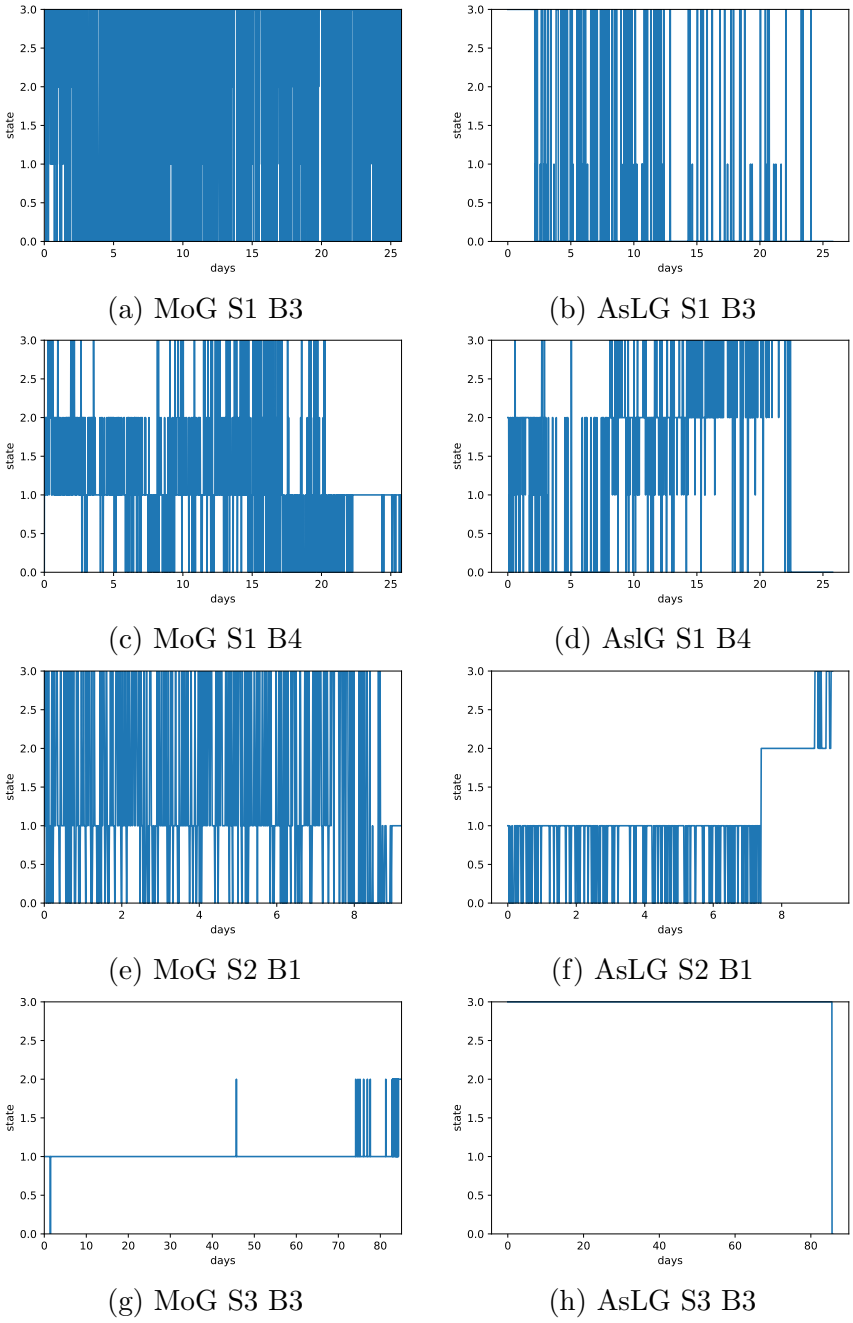
Table. 4.5 shows the results obtained from the MoG-HMM and AsLG-HMM when signal

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	1100	-101705.60	-105901.29	132	-70241.38	**-70677.90**
S1	B2	1100	-79674.65	-83870.33	129	-61950.20	**-62375.43**
S1	B3	1100	-109246.54	-113442.23	560	-76428.53	**-78475.65**
S1	B4	1100	-562364.09	**-566559.77**	134	nan	nan
S2	B1	1100	-16511.00	-20242.45	132	-9513.05	**-9891.67**
S2	B2	1100	-24369.24	-28100.69	129	-15488.07	**-15856.90**
S2	B3	1100	-25121.29	-28852.74	560	-19216.65	**-20992.26**
S2	B4	1100	-22605.34	-26336.79	134	-14049.49	**-14434.64**
S3	B1	1100	-85819.56	-90624.45	132	-69847.83	**-70354.06**
S3	B2	1100	-189836.81	-194641.70	129	-172699.36	**-173192.49**
S3	B3	1100	-621097.29	**-625902.18**	560	nan	nan
S3	B4	1100	-172836.71	-177641.61	134	-127218.05	**-127733.01**

Table 4.5: Likelihood and BIC results when S2 is used as training set with 3 harmonics.

S2 was used for learning the model parameters and three harmonics were considered. For this scenario, all the BIC scores were better for AsLG-HMM; however a numerical error raised when S1 B4 and S3 B3 were tested. With respect to the number of parameters, it was observed that in this case for AsLG-HMM, they had between 129 and 560 parameters, which, is greatly fewer than the 1100 parameters required by MoG-HMM. Here, again, it was observed the advantages of using asymmetric models: the number of parameters was adapted depending the data and unnecessary parameters were not estimated nor considered. Regarding the health estimations, they are drawn in Fig. 4.5. The health state sequence estimations obtained from AsLG-HMMs can be seen in (b), (d), (f) and (h). For S1 B3 (b) and S1 B4 (d) the sequences were noisy and not clear. However, in S2 B1 (f), it was observed an evolution of states where state 0 and 1 were almost equivalent and state 3 was the faulty state. For S3 B3 (h), a jump from state 3 to 0 was observed, which implies that the healthy state was 3 and the faulty state was 0. Meanwhile, the results obtained with MoG-HMM can be seen in (a), (c), (e) and (g). For S1 B3(a), S1 B4 (c) and S3 B3 (e), it is seen a noisy behavior which could not give any important information about the ball bearings health state. In the case of S3 B3 (h), it was observed a jump from state 1 to state 2 at the end of the bearing life which could imply that the failure state was 2 and the healthy state was 1.

Table. 4.6 shows the results obtained from MoG-HMM and AsLG-HMM when signal S3 was used for learning the models parameters and zero harmonics were considered. In this case the AsLG-HMM did not get the best BIC score for all the cases, specially when signals 3 was being tested. Regarding the number of parameters, the same behavior as in previous tests was observed: AslG-HMMs used lesser parameters than MoG-HMMs. The viterbi paths or health state estimations for this case, are pictured in Fig. 4.6. The state sequences obtained by the AsLG-HMMs are shown in (b), (d), (f) and (h). For S1 B3 (b), a constant health state was estimated, which is not true. For S1 B4 (d), a fast ball bearing health evolution was observed, where the faulty state 3 was reached in a short time. For S2 B1 (f), a jump from healthy state 0 to faulty state 3 was observed in the last part of the ball bearing operation time. For S3 B3 (h), an evolutionary health state was seen from the healthy state 0 to a



(a) MoG S1 B3 (b) AsLG S1 B3

(c) MoG S1 B4 (d) AslG S1 B4

(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.5: Sequences of states determined by the parameters learned from S2 with 3 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

faulty state 3. In comparison, the sequences obtained by MoG-HMM are shown in (a), (c), (e) and (g). For S1 B3 (a), a noisy and uninformative behavior was recorded. For S1 B4 (c) and S2 B1 (e), it is observed that the state 2 was the healthy one and state 0 was the faulty

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	92	-33280.40	-33631.31	44	-12174.38	**-12324.9**
S1	B2	92	-13487.54	-13838.45	44	-11484.06	**-11634.58**
S1	B3	92	-24326.31	-24677.22	44	-7202.71	**-7353.23**
S1	B4	92	-32105.84	-32456.75	44	-6446.02	**-6596.55**
S2	B1	92	-35808.70	-36120.79	44	-23150.37	**-23280.93**
S2	B2	92	-4503.32	**-4815.40**	44	-6437.78	-6568.34
S2	B3	92	-7928.76	-8240.85	44	-3936.90	**-4067.46**
S2	B4	92	-2292.98	-2605.07	44	-1908.85	**-2039.41**
S3	B1	92	-31088.49	**-31490.36**	44	-43653.76	-43828.32
S3	B2	92	-22319.74	**-22721.61**	44	-48086.59	-48261.15
S3	B3	92	nan	nan	44	-48625.15	**-48799.71**
S3	B4	92	-7777.07	**-8178.93**	44	-19073.07	-19247.64

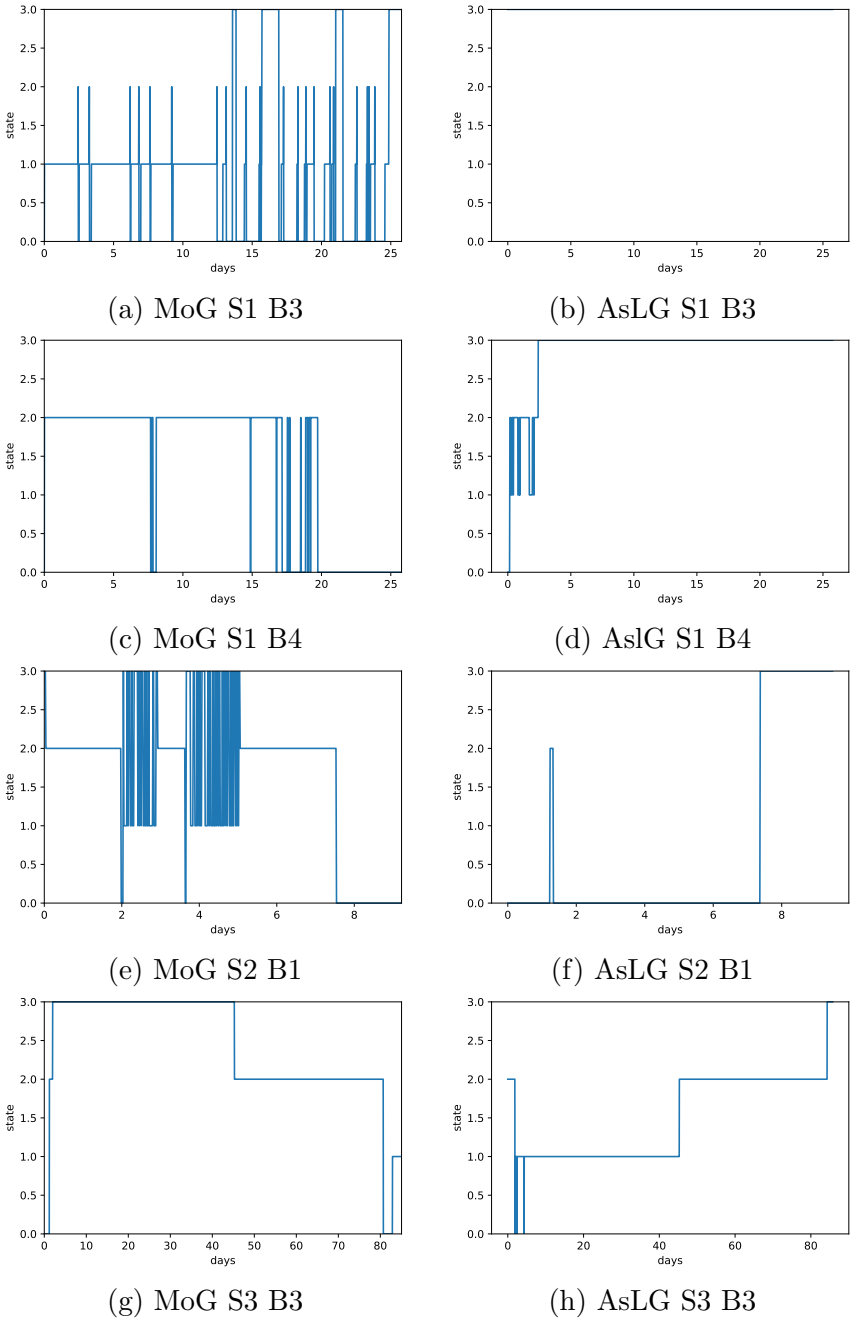
Table 4.6: Likelihood and BIC results when S3 is used as training set with 0 harmonics.

one; however, the faulty state is visited repeatedly at the begin of the ball bearing operation time, which is not true. For S3 B3 (g), an evolution from state 3 to state 0 was observed, which implies that 3 was the healthy state and 0 was the faulty state.

		MoG-HMM			AsLG-HMM		
Test	Bearing	#	LL	BIC	#	LL	BIC
S1	B1	1100	-104929.29	-109124.98	560	-48149.07	**-50196.19**
S1	B2	1100	-60061.17	-64256.86	560	-43337.75	**-45384.87**
S1	B3	1100	-99926.12	-104121.81	137	-32174.89	**-32630.22**
S1	B4	1100	-224770.40	-228966.09	133	-23468.56	**-23908.84**
S2	B1	1100	-127055.72	-130787.18	560	-48319.57	**-50095.18**
S2	B2	1100	-22662.44	-26393.89	560	-23117.11	**-24892.72**
S2	B3	1100	-48570.38	-52301.83	137	-16930.24	**-17325.18**
S2	B4	1100	-14156.58	-17888.03	133	-7751.82	**-8133.70**
S3	B1	1100	-140846.75	**-145651.65**	560	-162654.42	-165028.46
S3	B2	1100	-125170.86	**-129975.75**	560	-178040.19	-180414.23
S3	B3	1100	-34370.69	-39175.58	137	-270698.52	**-271226.57**
S3	B4	1100	-77744.42	-82549.32	133	-74495.96	**-75006.56**

Table 4.7: Likelihood and BIC results when S3 is used as training set with 3 harmonics.

Table. 4.7 shows the results obtained for the MoG-HMM and AsLG-HMM when signal S3 is used for learning the models parameters and three harmonics are considered. As in the case with no harmonics, AsLG-HMMs did not get the best BIC score for all cases, specially when S3 was being tested. The number of parameters for AsLG-HMM was always lower when compared with MoG-HMM, and such number could change with the ball bearing, giving insights of the dependency of the number of parameters with the data. Regarding the Viterbi paths or health estimations, Fig. 4.7 draws them. The health state sequence estimations obtained by the AsLG-HMM are pictured in (b), (d), (f) and (h) and the ones from MoG-HMM can be seen in (a), (c), (e) and (g). For S1 B3 (b) and S1 B4 (d), AsLG-HMM provided noisy estimations with little interpretation. For S2 B1 (f), a noisy behavior between the states 0, 1 and 2 was observed, but later, the health estimation was constant for state 3, which implies that states 0,1 and 2 were equivalent and state 3 was the faulty one. For S3 B3 (h), a noisy evolution of hidden states was observed, where 0 was the healthy



(a) MoG S1 B3 (b) AsLG S1 B3

(c) MoG S1 B4 (d) AslG S1 B4

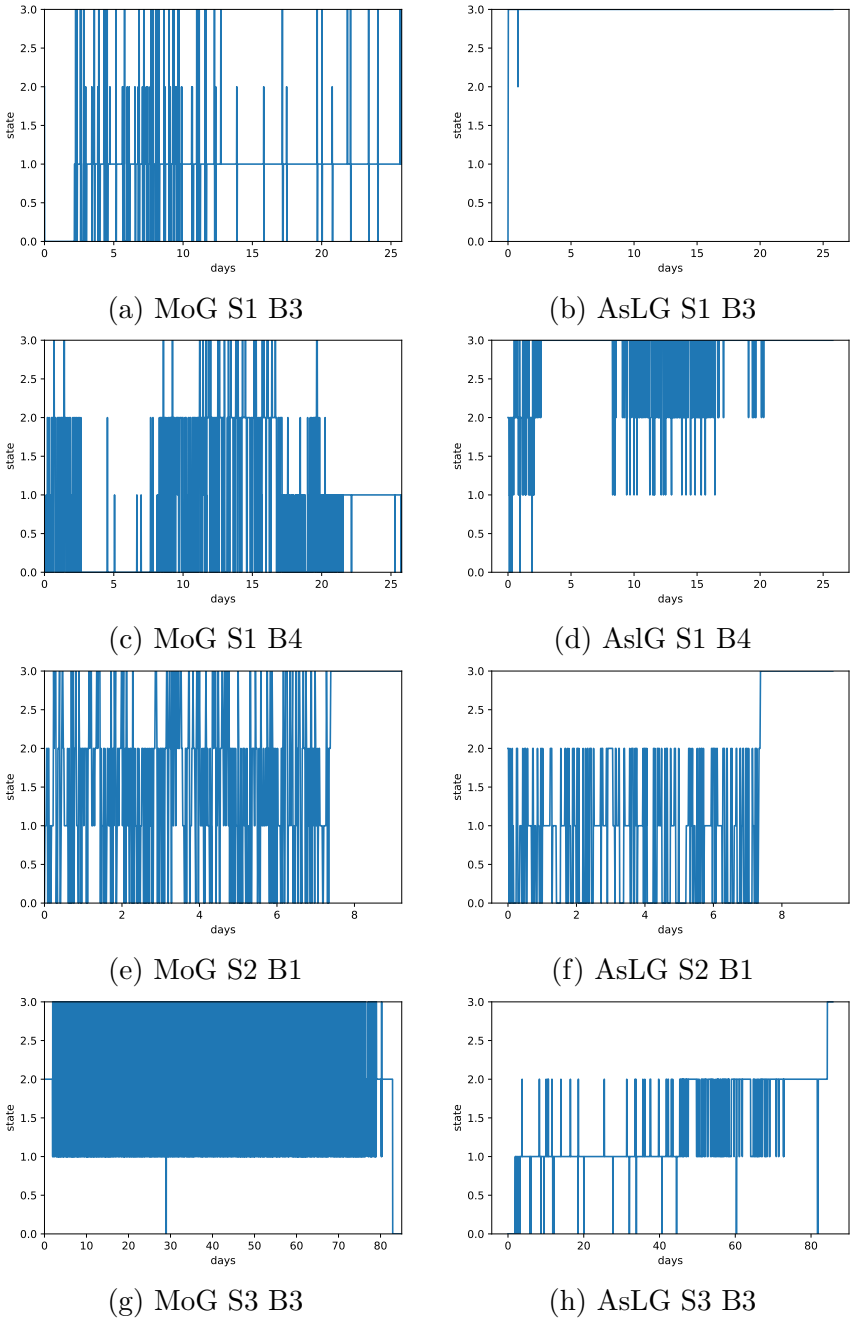
(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.6: Sequences of states determined by the parameters learned from S3 with 0 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

state and 3 was the faulty state, 1 and 2 were intermediate health states. For the case of MoG-HMM, for all the literals, a noisy with little interpretation behavior was observed.

In Fig. 4.8, it can be seen some of the structures learned by the proposed model when 0



(a) MoG S1 B3 (b) AsLG S1 B3

(c) MoG S1 B4 (d) AslG S1 B4

(e) MoG S2 B1 (f) AsLG S2 B1

(g) MoG S3 B3 (h) AsLG S3 B3

Figure 4.7: Sequences of states determined by the parameters learned from S3 with 3 harmonics. (a) MoG-HMM sequence of states of S1 B3. (b) AslG-HMM sequence of states of S1 B3. (c) MoG-HMM sequence of states of S1 B4. (d) AslG-HMM sequence of states of S1 B4. (e) MoG-HMM sequence of states of S2 B1. (f) AslG-HMM sequence of states of S2 B1. (g) MoG-HMM sequence of states of S4 B3. (h) AslG-HMM sequence of states of S4 B3.

harmonics were used. Only state 0 (healthy) and state 3 (faulty) are shown for the case of the known faulty bearings. These structures give us light of the relationship between variables. For example, (a) shows that for this state BPFI had no parents and all other variables were

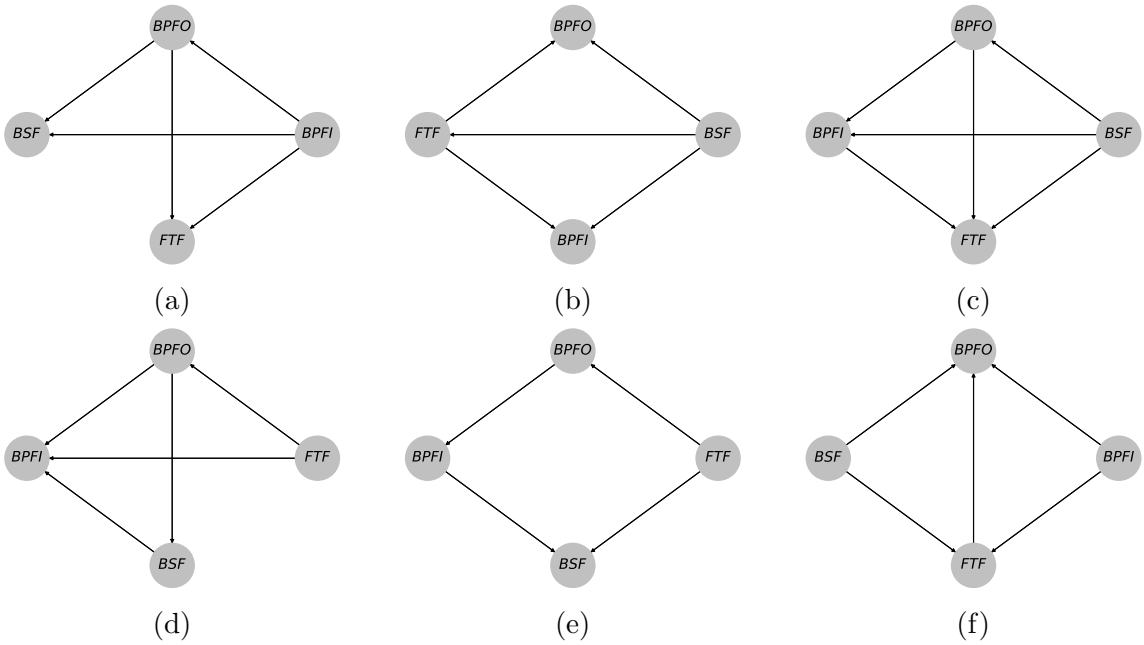


Figure 4.8: Structures learned for the case where 0 harmonics were used. (a) is the state 0 from S1 B3. (b) is the state 3 from S1 B3. (c) is the state 0 from S1 B4. (d) is the state 3 from S1 B4. (e) is the state 0 from S2 B1. (f) is the state 3 from S2 B1.

dependent to it, therefore the frequential behaviour was due to this frequency for state 0. On the other hand, (b) shows that for state 3, the BSF was responsible of the behaviour of the other 3 frequencies. Note that BPFO nor BPFI had sons, which implies that they were totally determined by BSF and FTF frequencies. It is possible to make this kind of interpretations to the remaining models and understand which variables were determining or not the dynamic behaviour of the tool.

It can said that in general, the signal S1 was better to model the dynamic behavior of the ball bearings, specially when no harmonics were considered. When harmonics were taken into consideration for the models, the health estimations were noisy and could lost interpretation. In spite of this, in Chapter 6 and Chapter 9, offline and online models with feature selection will be introduced in order to filter harmonics and extract additional insights from data.

4.4 Conclusions

In this chapter the AsLG-HMM has been introduced in order to deal with continuous variables in asymmetric HMMs. This model was proposed to overcome data overfitting and discretization steps. Also, AsLG-HMMs are capable of providing useful interpretation of the problem domain, since state-specific LGBNs are used. Additionally, since the number of parameters are reduced when compared to other models such as MoG-HMMs, inference tasks are easier to perform.

In the numerical experiments it was seen that AsLG-HMMs obtained in general better BIC scores than those from MoG-HMMs. Also, the ball bearing health estimation sequences from AsLG-HMMs were easier to interpret and were more informative than in the case of MoG-HMMs. Finally, it was observed that when no harmonics were used, the model predictions were more stable and less noisy, which was desirable for interpretation and data insights extraction.

In spite of these first results, there are still some issues to solve such as the interpretability of the Viterbi paths, which can be unreliable and hard to measure. The execution times of the simulated annealing could be large, even for small datasets. Therefore, another search method must be applied in order to have shorter training times and stable results. Finally, it is interesting for industrial processes and for specifically, ball bearings, to determine the health and which frequencies are relevant over their operation time in data streams environments. Most issues will be addressed with the models introduced in the next chapters.

Chapter 5

Autoregressive AsLG-HMMs

In this chapter, the AslG-HMM introduced in Chapter 4 is extended by introducing an AR component to each variable. The AR order of each variable for each possible context is determined by the SEM algorithm and the Yule-Walker equations when a score (to be specified later on) is optimized. A greedy-forward heuristic algorithm is introduced to search in the DAG space in order to obtain more stable and shorter training times. Also, a hidden state labeling function is defined to ease the HMM interpretation and data insight extraction. Furthermore, for the proposed model, the likelihood function is modified; therefore, the forward-backward, Viterbi and EM algorithms have to be adapted. The main contents of this is in Puerto-Santana et al. [2022b]. Synthetic and real data from air pollution and ball bearing degration are used for validation. The proposed model is contrasted with several models based on HMMs in order to prove the benefits from taking into consideration asymmetric and AR behaviors.

The chapter is organized as follows: Section 5.1 introduces the new model based on AsLG-HMMs. Section 5.2 explains how to adapt the traditional EM algorithm to the proposed model. Section 5.3 adapts the forward-backward algorithm to the new model hypotheses. Section 5.4 explains how to apply the adapted EM algorithm to learn the new model parameters. Section 5.5 extends the traditional Viterbi algorithm to the proposed model. Section 5.6 indicates the new restrictions and considerations to apply the SEM algorithm and also a new greedy forward algorithm is introduced for the graph search. Section 5.7 introduces a state labeling function to identify magnitude changes in HMMs. Section 5.8 shows the results of applying the proposed model for different synthetic and real-world datasets. Finally, in Section 5.9 the conclusions and comments related to the results and innovations from the chapter are provided.

5.1 Model proposal

Let p_m^* be the AR order (time lag) determined by the Yule-Walker equations (see Section 2.5.3) and the individual relevancy hypothesis tests for each variable X_m^t, $m = 1, ..., M$.

Set $p^* = \max_m p_m^*$. For our proposed mode, the following log-likelihood function is considered, which ensures that during the SEM algorithm the updated structures and AR orders are comparable:

$$\begin{aligned} LL(\boldsymbol{\lambda}) &= \ln P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \\ &= \ln \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}). \end{aligned} \tag{5.1}$$

For this proposed HMM model which is, as explained below, asymmetric autoregressive with linear Gaussian emission probabilities (AR-AsLG-HMM), the emission probabilities $\{b_i(\boldsymbol{x}^t)\}_{i=1}^N$ are modified such that they can be factorized into linear Gaussian Bayesian networks with an asymmetric component, i.e., each variable X_m for each state $i \in R(Q)$ is associated with a set of parents $\mathbf{Pa}_i^t(X_m) = \{U_{im1}^t, ..., U_{imk_{im}}^t\} \subset \{X_1^t, ..., X_M^t\}$ of size k_{im} (apart from Q) which influences its mean in a linear form. Additionally, the emission probabilities are now conditional probabilities given $p_{im} \leq p^*$ past values of the variables X_m^t, $m = 1, ..., M$ (AR terms) for each state $i \in R(Q)$. More specifically, the emission probabilities are:

$$\begin{aligned} b_i^{p^*}(\boldsymbol{x}^t) &= P(\boldsymbol{x}^t|Q^t = i, \boldsymbol{x}^{t-p^*:t-1}, \boldsymbol{\lambda}) \\ &= \prod_{m=1}^{M} P(x_m^t|Q^t = i, x_m^{t-p_{im}:t-1}, \mathbf{Pa}_i(X_m), \boldsymbol{\lambda}) \\ &= \prod_{m=1}^{M} \mathcal{N}(x_m^t|\boldsymbol{u}_{im}^t\boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t\boldsymbol{\eta}_{im}, \sigma_{im}^2) \end{aligned} \tag{5.2}$$

In Eq. (5.2), $\boldsymbol{\beta}_{im} = (\beta_{im0}, ..., \beta_{imk_{im}})^\top$, $\boldsymbol{u}_{im}^t = (1, u_{im1}^t, ..., u_{imk_{im}}^t)$, $\boldsymbol{\eta}_{im} = (\eta_{im1}, ..., \eta_{imp_{im}})^\top$ and $\boldsymbol{d}_{im}^t = (x_m^{t-1}, ..., x_m^{t-p_{im}})$. Fig. 5.1 shows an example of an AR-AsLG-HMM. In this example, when $Q^t = 1$ (top), variable X_2^t is dependent on Q^t, X_1^t, X_2^{t-1} and X_2^{t-2}, but X_1^t depends only on Q^t and X_1^{t-1}. However, when $Q^t = 2$ (bottom), X_1^t depends only on Q^t and X_2^t is dependent on X_2^{t-1} and Q^t. In terms of the model, this can be expressed as $p_{11} = 1$, $p_{12} = 2$ AR terms, $k_{11} = 0$ and $k_{12} = 1$ when $Q = 1$, and $p_{21} = 0$, $p_{22} = 1$ AR terms, $k_{21} = 0$ and $k_{22} = 0$ when $Q = 2$. From the model it can be seen that $p^* \geq 2$, because $p_{im} \leq 2$ for $i = 1, 2$ and $m = 1, 2$.

Some comments on Eq. (5.2) follows: the set of parents $\mathbf{Pa}_i^t(X_m)$ of each variable X_m for each state $i \in R(Q)$ is related to a context-specific Bayesian network graph $\mathcal{B}_i$. Furthermore, depending on that hidden state, each variable X_m may have a different AR order, namely p_{im}, which is upper bounded by p^*. This model must estimate the parameters $\{\beta_{im0}, ..., \beta_{imk_{im}}, \eta_{im1}, ..., \eta_{imp_{im}}, \sigma_{im}^2\}_{m=1,i=1}^{M,N}$. Additionally, because the first p^* observations are used as conditionals in Eq. (5.1), the $\boldsymbol{\pi}$ parameter is shifted to predict the initial distribution of the Q^{p^*} hidden variable, i.e., $\{\pi_i\}_{i=1}^N = \{P(Q^{p^*} = i|\boldsymbol{\lambda})\}_{i=1}^N$. Observe that the complete information probability of an instance $\boldsymbol{x}^{p^*:T}$ of $\boldsymbol{X}^{p^*:T}$ and an instance $\boldsymbol{q}^{p^*:T}$ of $\boldsymbol{Q}^{p^*:T}$

can be expressed as:

$$P(\boldsymbol{q}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) = \pi_{q^{p^*}} \prod_{t=p^*}^{T-1} a_{q^t q^{t+1}} \prod_{t=p^*}^{T} b_{q^t}^{p^*}(\boldsymbol{x}^t).$$

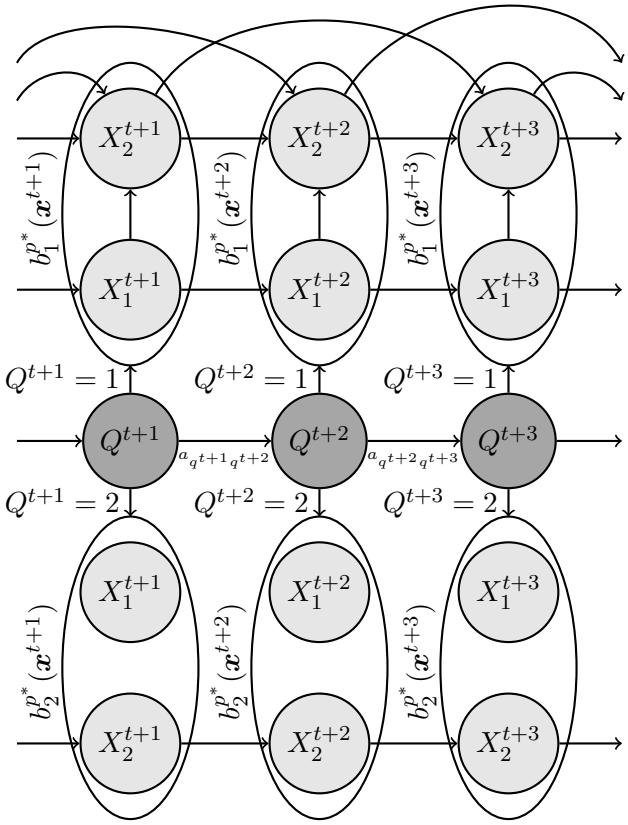


Figure 5.1: Graphical representation of an AR-AsLG-HMM model

5.2 Feasibility of the EM algorithm in AR-AsLG-HMMs

To perform the parameter learning, the EM algorithm can be applied. However, it must be defined an auxiliary function $\mathcal{Q}$ for the log-likelihood defined in Eq. (5.1). It is proposed $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ as the auxiliary function for the EM algorithm, defined as:

$$\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}). \tag{5.3}$$

Note that $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ can be decomposed as:

$$\begin{aligned}\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) &= \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}) \\ &\quad + \ln P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1},\boldsymbol{\lambda}) \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}') \\ &= \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}) + LL(\boldsymbol{\lambda})\end{aligned} \tag{5.4}$$

If $\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ is defined as the first summand of Eq. (5.4), i.e.,

$$\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) := \sum_{\boldsymbol{R}(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}) \tag{5.5}$$

therefore: $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + LL(\boldsymbol{\lambda})$, similar to Eq. (2.24). Below, it is shown that if the EM algorithm is applied with $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$, each iteration does not decrease the log-likelihood as required. Or in other words, the EM algorithm in Dempster et al. [1977] can be extended to the proposed likelihood function.

Lemma 5.1. Let $\boldsymbol{\lambda}^{(s)}$ be the parameters at iteration s of the EM and $\boldsymbol{\lambda}^{(s+1)}$ be the resulting parameters after the next iteration of the EM. The following follows: $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) \geq \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)})$.

■

Lemma 5.2. Given two arbitrary models with respective parameters $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$, the following follows $\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') \leq \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}')$, and the equality holds when $P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}) = P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}')$.

■

Theorem 5.1. Let $\boldsymbol{\lambda}^{(s)}$ be the parameters at an iteration s of the EM and $\boldsymbol{\lambda}^{(s+1)}$ be the resulting parameters after the next iteration of the EM. The following holds:

(a) $LL(\boldsymbol{\lambda}^{(s+1)}) \geq LL(\boldsymbol{\lambda}^{(s)})$. In other words, the log-likelihood of the model cannot worsen after an EM iteration.

(b) The sequence $\{LL(\boldsymbol{\lambda}^{(s)})\}_{s\in\mathbb{N}}$ converges.

■

The proofs of the lemmas and theorems can be found in the Appendix A.

5.3 The forward-backward algorithm in AR-AsLG-HMMs

As the likelihood function of Eq. (5.1) and the emission probabilities given by Eq. (5.2) have changed, the forward-backward algorithm must be adapted. In the E step, it is necessary to

compute the probabilities $\gamma^t(i) = P(Q^t = i|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda})$ for $t = 0, ..., T$ and $i = 1, ..., N$ as the initial point to fit the forward-backward algorithm. Note that $\gamma^t(i)$ can be expressed as:

$$\begin{aligned}
\gamma^t(i) &= \frac{P(Q^t = i, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})}{P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})} \\
&= \frac{P(Q^t = i, \boldsymbol{x}^{p^*:t}, \boldsymbol{x}^{t+1:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})}{P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})} \\
&= \frac{P(\boldsymbol{x}^{t+1:T}|Q^t = i, \boldsymbol{x}^{0:t}, \boldsymbol{\lambda})P(Q^t = i, \boldsymbol{x}^{p^*:t}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})}{P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})} \\
&= \frac{\beta^t_{p^*}(i)\alpha^t_{p^*}(i)}{\sum_{j=1}^N \beta^t_{p^*}(j)\alpha^t_{p^*}(j)}
\end{aligned} \tag{5.6}$$

From Eq. (5.6), the forward variable is $\alpha^t_{p^*}(i) := P(Q^t = i, \boldsymbol{x}^{p^*:t}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})$ and the backward variable is $\beta^t_{p^*}(i) := P(\boldsymbol{x}^{t+1:T}|Q^t = i, \boldsymbol{x}^{0:t}, \boldsymbol{\lambda})$. Observe that these equations only make sense when $t \geq p^*$ and differ slightly from those in Section 2.3.3. The next lemma shows that the traditional forward-backward algorithm can be adapted to compute the α_{p^*} and β_{p^*} parameters of an AR-AsLG-HMM.

Lemma 5.3. $\alpha^t_{p^*}(i)$ and $\beta^t_{p^*}(i)$ can be computed as:

$$\begin{aligned}
\alpha^t_{p^*}(i) &= \sum_{j=1}^N b_i^{p^*}(\boldsymbol{x}^t)a_{ji}\alpha^{t-1}(j) \\
\beta^t_{p^*}(i) &= \sum_{j=1}^N \beta^{t+1}(j)b_j^{p^*}(\boldsymbol{x}^{t+1})a_{ij}
\end{aligned} \tag{5.7}$$

for $t = p^*, ..., T$ and $i = 1, ..., N$, with initial values $\alpha^{p^*}_{p^*}(i) = \pi_i b_i^{p^*}(\boldsymbol{x}^{p^*})$ and $\beta^T_{p^*}(i) = 1$, $i = 1, ..., N$.

■

5.4 Parameter learning in AR-AsLG-HMMs

To execute the EM algorithm, the E step and the M step must be iterated. For the E step it can be used the adapted forward-backward algorithm of Section 5.3 to compute $\gamma^t(i)$ and $\xi^t(i,j)$ for $i, j = 1, ..., N$ and $t = 0, ..., T$:

$$\begin{aligned}
\gamma^t(i) &= \frac{\beta^t_{p^*}(i)\alpha^t_{p^*}(i)}{\sum_{j=1}^N \beta^t_{p^*}(j)\alpha^t_{p^*}(j)} \\
\xi^t(i,j) &= \frac{\alpha^t_{p^*}(i)a_{ij}b_j^{p^*}(\boldsymbol{x}^{t+1})\beta^{t+1}_{p^*}(j)}{\sum_{u,v=1}^N \alpha^t_{p^*}(u)a_{uv}b_v^{p^*}(\boldsymbol{x}^{t+1})\beta^{t+1}_{p^*}(v)}
\end{aligned} \tag{5.8}$$

Computing these quantities is enough for the E step because $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ can be expressed as:

$$\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \gamma^{p^*}(i) \ln \pi_i + \sum_{t=p^*}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi^t(i,j) \ln a_{ij} + \sum_{t=p^*}^{T} \sum_{i=1}^{N} \gamma^t(i) \ln b_i^{p^*}(\boldsymbol{x}^t). \tag{5.9}$$

For the M step, the updating formulas for the parameters $(\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ must be deduced. In the following theorem, it is provided the updating formulas for the proposed model.

Theorem 5.2. The M-step for an AR-AsLG-HMM model can be performed using the following updating formulas: parameter $\boldsymbol{\pi} = \{\pi_i\}_{i=0}^N$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{5.10}$$

The parameter $\mathbf{A} = \{a_{ij}\}_{i,j=1}^N$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{5.11}$$

If $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$, the parameters $\{\eta_{imr}\}_{r=1}^{p_{im}}$, $\{\beta_{imk}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear system:

$$\begin{cases} \sum_{t=p^*}^{T} \gamma^t(i) x_m^t = \sum_{t=p^*}^{T} \gamma^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^{T} \gamma^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^{T} \gamma^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \gamma^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^{T} \gamma^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^{T} \gamma^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^{T} \gamma^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \gamma^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^{T} \gamma^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{5.12}$$

if $\boldsymbol{\theta}_{im} = (\boldsymbol{\beta}_{im}|\boldsymbol{\eta}_{im})^\top$, $\boldsymbol{o}_{im}^t = (\boldsymbol{u}_{im}^t|\boldsymbol{d}_{im}^t)$, and $\boldsymbol{\Gamma}_i^{p^*:T} := \text{Matrix}([\gamma^{p^*}(i), ..., \gamma^T(i)])$ Then, the previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{5.13}$$

If $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$, then, σ_{im}^2 can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^{T} \gamma^t(i)(x_m^t - \hat{\varphi}_{im}^t)^2}{\sum_{t=p^*}^{T} \gamma^t(i)}. \tag{5.14}$$

This update must be done for every variable $m = 1, ..., M$ and hidden state $i = 1, ..., N$.

■

Eq. (5.12) forms a linear system of $k_{im} + p_{im} + 1$ unknowns with $k_{im} + p_{im} + 1$ equations. If the resulting context-specific Bayesian model for every hidden state is a naïve Bayesian network and $p_{im} = 0$ for $i = 1, ..., N$ and $m = 1, ..., M$, then it is only required to update the parameters $\{\beta_{im0}\}_{i=1,m=1}^{N,M}$. Its updating formula is:

$$\beta_{im0}^{(s+1)} = \frac{\sum_{t=p^*}^{T} \gamma^t(i) x_m^t}{\sum_{t=p^*}^{T} \gamma^t(i)}.$$

Following the discussion in Section 4.2, the complexity of updating the parameters $\boldsymbol{\theta} = \{\theta_{im}\}_{i=1,m=1}^{N,M}$ for the worst case scenario is $O(NM(M^3 + {p^*}^3))$. In such scenario, all the possible arcs for a DAG are present and every variable has p^* AR dependencies. This situation is highly undesirable, since the computational cost can increase rapidly with the number of variables and p^*. However, this kind of scenarios, again, can be avoided during the SEM due to the penalization of structures with several parameters.

5.5 The Viterbi algorithm in AR-AsLG-HMMs

In the following lemma, it is shown that the traditional Viterbi algorithm can be adapted to determine the most probable sequence of hidden states in AR-AsLG-HMMs.

Lemma 5.4. If $\delta_{p^*}^t(i) = \max_{\boldsymbol{q}^{p^*:t-1}}\{P(\boldsymbol{x}^{p^*:t}, \boldsymbol{q}^{p^*:t-1}, Q^t = i|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})\}$ represents the most probable sequence of hidden states up to time $t-1$ for state i at time t, then $\delta_{p^*}^t(i)$ can be computed recursively.

$$\delta_{p^*}^t(i) = \max_{j=1,...,N}\{\delta_{p^*}^{t-1}(j)a_{ji}\}b_i^{p^*}(\boldsymbol{x}^t)$$

The Viterbi algorithm is initialized with $\delta_{p^*}^{p^*}(i) = \pi_i b_i^{p^*}(\boldsymbol{x}^{p^*})$.

■

5.6 The SEM algorithm in AR-AsLG-HMMs

Regarding the structural optimization process, the SEM algorithm for AR-AsLG-HMMs must also be modified. The proposed auxiliary function is:

$$\mathcal{Q}^{p^*}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(s)}, \boldsymbol{\lambda}^{(s)}) = \mathbb{E}_{P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \mathcal{B}^{(s)}, \boldsymbol{\lambda}^{(s)})} \ln P(\boldsymbol{x}^{p^*:T}, \boldsymbol{Q}^{p^*:T}|\mathcal{B}, \boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) - 0.5\#(\mathcal{B})\ln(T). \tag{5.15}$$

The steps for the adapted SEM algorithm are the same as in the general SEM. However, it must considered that given a time slice t, the algorithm must not only look for the best instantaneous structure at time t or the best structure with variables $(X_1^t, ..., X_M^t)$ but also look for the best transition structure at time t or the relationships between $(X_1^t, ..., X_M^t)$ variables and their AR versions, i.e., $(X_1^{t-1}, X_1^{t-2}, ..., X_M^{t-p^*+1}, X_M^{t-p^*})$, which implies that the search space dimension increases. More specifically, it must be explored not only the space of directed acyclic graphs (DAGs) for the best instantaneous structures, but also in the space

$S_{p^*} = \{0, 1, ..., p^*\}^N \times \{0, 1, ..., p^*\}^M$, for the best transition structure. A component p_{im} of a matrix $\boldsymbol{p} \in S_{p^*}$ indicates the number of lags for variable X_m in the hidden state $i \in R(Q)$. For instance, if $p_{im} = 2$, X_m^t has incoming arcs from the variables X_m^{t-2} and X_m^{t-1} when $Q^t = i$.

It is pertinent to mention that in the SEM algorithm in the step of finding $\mathcal{B}^{(s)} = \arg\max_{\mathcal{B}} \mathcal{Q}^{p^*}(\mathcal{B}, \boldsymbol{\lambda}^{(s)} | \mathcal{B}^{(s-1)}, \boldsymbol{\lambda}^{(s)})$ it is not necessary to use Eq. (5.15), since the initial distribution and the transition matrix are kept unchanged. It is possible to take advantage of the linearity of Eq. (5.15) to compare structures, i.e., if a dependency of X_m has been added or deleted (AR or parent parameter) at the hidden state i, it is reasonable to use the following score:

$$\text{score}_{im} = \sum_{t=p^*}^{T} \gamma^t(i) \ln(\mathcal{N}(x_m^t | \varphi_{im}^t, \sigma_{im}^2)) - 0.5\#(\mathcal{B}_i) \ln(T+1). \tag{5.16}$$

If changes have been done to many variables in many hidden states, it is better to use the following score:

$$\text{score} = \sum_{i=1}^{N} \sum_{m=1}^{M} \text{score}_{im}. \tag{5.17}$$

To perform the structural optimization step, it is explored the space of structures. In this chapter, a heuristic forward greedy algorithm is proposed to perform the structure optimization. In this approach, all the structures are started as naïve Bayesian networks with no AR parameters. During the optimization, each variable is visited for each hidden state and it is added AR or parent dependencies as long as Eq. (5.16) improves. Its pseudocode is shown in Fig. 5.1.

Other algorithms have been used to search in the graph space during the SEM algorithm, e.g., Bueno et al. [2017] used a tabu search algorithm (see,Glover and Laguna [1997]), and in Chapter 4, it was used a simulated annealing algorithm. In general, any meta-heuristic or heuristic can be used to search in the space of DAGS.

5.7 Hidden state labelling

In practice, when HMMs are used, categorical labels are given to the hidden states for interpretation purposes as done in Section 4.3.3. However, only after training the model, the model parameters are manually checked to determine which categorical label corresponds with each trained hidden state. Here, it is proposed an automatic numerical labelling for trained models, where a numerical function is used to label a trained hidden state. Let $g : R(Q) \to \mathbb{R}$ be a function that maps each hidden state into a real number depending on the models parameters. This function g not only helps us determine whether a change in hidden states occurs but also the magnitude of the change. For example, if deviations from known standards or desired values $\boldsymbol{\kappa} = \{\kappa_1, ..., \kappa_m\}$ of $\boldsymbol{X}$ imply changes in state, the following g functions described in Eq. (5.18) and Eq. (5.19) can be used to help in hidden

Algorithm 5.1 Pseudo-code for the forward greedy algorithm

Input: A parameter $\boldsymbol{\lambda}^{(s)}$, a prior structure $\mathcal{B}^{(s-1)}$
Return: A structure $\mathcal{B}^{(s)}$
  Set $\mathcal{B}^{(s)} := \mathcal{B}^{(s-1)}$
  ** *Optimization of AR structures* **
  For $i = 1, ..., N$ **do**:
    For $m = 1, ..., M$ **do**:
      Compute score_{im} with $\mathcal{B}^{(s-1)}$ and $\boldsymbol{\lambda}^{(s)}$
      While $p_{im} \leq p^*$ **do**:
        Define $\hat{\mathcal{B}}$ with $\hat{p}_{im} := p_{im} + 1$ and estimate $\widehat{\boldsymbol{\lambda}} = \arg\max_{\boldsymbol{\lambda}} \mathcal{Q}^{p^*}(\widehat{\mathcal{B}}, \boldsymbol{\lambda} | \mathcal{B}^{(s-1)}, \boldsymbol{\lambda}^{(s)})$
        Compute $\widehat{\text{score}_{im}}$ with $\widehat{\mathcal{B}}$ and $\widehat{\boldsymbol{\lambda}}$
        If $\widehat{\text{score}_{im}} > \text{score}_{im}$ **then**:
          Update $\mathcal{B}^{(s)} := \hat{\mathcal{B}}$, $\boldsymbol{\lambda}^{(s)} := \hat{\boldsymbol{\lambda}}$ and $\text{score}_{im} := \max\{\text{score}_{im}, \widehat{\text{score}_{im}}\}$
        Else:
          Break
  ** *Optimization of instantaneous structures* **
  For $i = 1, ..., N$ **do**:
    For $m = 1, ..., M$ **do**:
      Compute $\boldsymbol{\mathcal{B}_{im}}$:= all the possible DAG graphs resulting by adding one arc to $\mathcal{B}^{(s)}$ in $\mathcal{B}_{im}$, where X_m is a new children of any variable
      If $\boldsymbol{\mathcal{B}_{im}}$ is non-empty **then**:
        Compute score_{im} with $\mathcal{B}^{(s)}$ and $\boldsymbol{\lambda}^{(s)}$
        For $\widehat{\mathcal{B}}$ in $\boldsymbol{\mathcal{B}_{im}}$
          Estimate $\widehat{\boldsymbol{\lambda}} = \arg\max_{\boldsymbol{\lambda}} \mathcal{Q}^{p^*}(\widehat{\mathcal{B}}, \boldsymbol{\lambda} | \mathcal{B}^{(s-1)}, \boldsymbol{\lambda}^{(s)})$
          Compute $\widehat{\text{score}_{im}}$ with $\hat{\mathcal{B}}$ and $\widehat{\boldsymbol{\lambda}}$
          If $\widehat{\text{score}_{im}} > \text{score}_{im}$ **do**:
            Update $\mathcal{B}^{(s)} := \hat{\mathcal{B}}$, $\boldsymbol{\lambda}^{(s)} := \hat{\boldsymbol{\lambda}}$ and $\text{score}_{im} := \max\{\text{score}_{im}, \widehat{\text{score}_{im}}\}$

states labelling in AR-AsLG-HMMs:

$$g_1(i) = \sum_{m=1}^{M} v_m(\nu_{im} - \kappa_m) \tag{5.18}$$

$$g_2(i) = \max_{m=1,...,M} \{v_m(\nu_{im} - \kappa_m)\} \tag{5.19}$$

Where

$$\nu_{im} = \frac{\beta_{im0} + \beta_{im1}\nu_{iu_{im1}} + \cdots + \beta_{imk_{im}}\nu_{iu_{imk_{im}}}}{1 - (\eta_{im1} + \cdots + \eta_{imp_{im}})}. \tag{5.20}$$

Observe that in Eq. (5.20), the value of ν_{im} depends on the ν value of the parents of variable X_m in the context-specific graph related to the i state, so Eq. (5.18) and Eq. (5.19) must be calculated recursively. The recursion begins with those variables that fulfil the following condition: $\mathbf{Pa}_i(X_m) = \emptyset$. Next, the recursion is computed for their descendants in the context-specific graph, until no variables are left. In general, ν_{im} can be interpreted as the mean of variable X_m at the hidden state i. Additionally, the vector $\mathbf{v} = (v_1, ..., v_M)$ for

$m = 1, .., M$ can be considered a feature relevance constant vector or a scaling constant vector that can be tuned according to the nature of the problem.

Eq. (5.18) can be used in cases where the addition of errors determines the driven process. For example, in the case of a country economy where the aggregation of economic variables can determine if there is economic growth or not. Or in the case of bearings degradation, where the aggregation of the amplitude of desired frequencies represents the presence of failure. On the other hand, Eq. (5.19) can be used when high deviations from a single variable is enough to determine the dynamical process. For example, consider a patient with a chronic disease with many sensors that measure different biological variables. For each variable there is a desirable value that determines good health. If only one variable drifts from the desirable value, the health of the patient can be in danger. In conclusion, the experiment and context of the problem may require a different g function to describe the hidden states.

5.8 Experimental setup

In this section, the proposed model (AR-AsLG-HMMs) is compared with AsLG-HMMs, LMSAR, AR-MoG-HMMs, MoG-HMMs, VAR-MVGHMMs, BMMs and a simple AR-AsLG-HMM with naïve Bayes context-specific Bayesian networks that will be called naïve-HMMs (this kind of models have been used in **?**). See Table 3.1 to see the respective reference for each model. In the case of LMSAR Hamilton [1990][1] and AR-MoG-HMM Juang and Rabiner [1985], it was defined only for one variable. Therefore, in these experiments, it is assumed that for these models every variable is independent. In particular, LMSAR is a special case of an AR-AsLG-HMM, where only AR parameters are used in the mean, but the number of them do not change with the hidden state, and the variance of the model does not depend on the hidden state. In the case of AR-MoG-HMM, the model assumes that the variances are unitary and do not depend on the hidden state; in spite of that it cannot be expressed directly as an AR-AsLG-HMM. Also, both AR-AsLG-HMM and AsLG-HMM use the forward-greedy algorithm in the SEM algorithm to ensure reproducibility. The aim is to show the capabilities of the proposed model to change the number of AR parameters and the context-specific Bayesian networks when they are needed.

Experiments with synthetic data are performed. The data are generated such that over time, the AR process changes. Two dynamic processes are used with six variables. The models are learned using only one time series, where three possible hidden states are present and appear in time blocks. The aim is to determine for new data the most likely sequence of hidden states. This sequence tells us the current probabilistic distribution of the data and therefore which probabilistic relationships are relevant. Also, the number of parameters and BIC score play an important role to identify which model is better as explained in Section 5.6.

Air quality data and real ball bearing degradation data are also used. The p^* values are computed using the Yule-Walker equations (see Section 2.5.3). They are used as well for AR-MoG-HMM, BMM, VAR-MVGHMM and LMSAR to determine the maximum lag during the

[1] See page 57

learning task. For the mixture models, two and three mixture components were used, and the models with two mixture components had in general the highest BIC and log-likelihood. Then, just two mixture components were used.

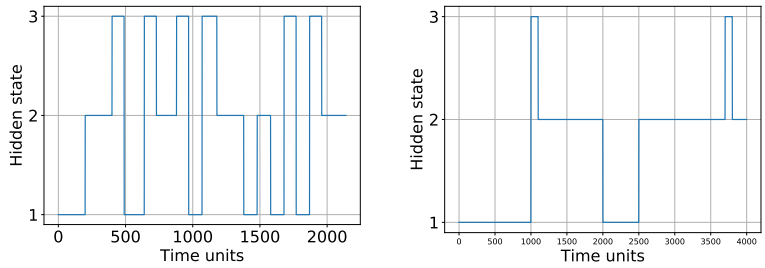
For both synthetic and real data, the models are initialized with a uniform transition matrix $\mathbf{A}$, specifically $a_{ij} = 1/N$; for $i, j = 1, ..., N$; the same for the initial distribution $\boldsymbol{\pi}$, specifically, $\pi_i = 1/N$ for $i = 1, ..., N$. In the case of the proposed AR-AsLG-HMM, the partial correlation function is evaluated up to five AR values to prevent high computational times and set $p_{im} = 0$ for $i = 1, ..., N$ and $m = 1, ..., M$; this means that no AR relationships are assumed a priori in the models. For both AR-AsLG-HMM and AsLG-HMM, all the context-specific Bayesian networks are initialized as naïve Bayes networks. The emission probabilities for the AR-AsLG-HMM and AsLG-HMM are initialized with $\beta_{im0} = i(\max_t x_m^t - \min_t x_m^t)/(N + 1) + \min_t x_m^t$ and $\sigma_{im}^2 = 2(\max_t x_m^t - \min_t x_m^t)$ for $i = 1, ..N$ and $m = 1, ..., M$. The purpose of this selection for β_{im0} is to initialize the mean of each variable for each hidden state in an equally separated different point in the possible range of values given by the training dataset. The selection of σ_{im}^2, is to avoid infinite or nan values in the first iterations of the forward-backward algorithm. For the mixture models, the distribution of the mixture coefficients is uniform, and the mean coefficients for the mixtures models are initialized using a k-means algorithm of clustering.

All the models were implemented in python 3 and the used libraries were numpy, matplotlib, pandas, networkx, math, pickle and the k-means algorithm from the scikit-learn library. No parallelization or own created external functions or libraries (like in C or C++) to improve the performance were used.

5.8.1 Synthetic data

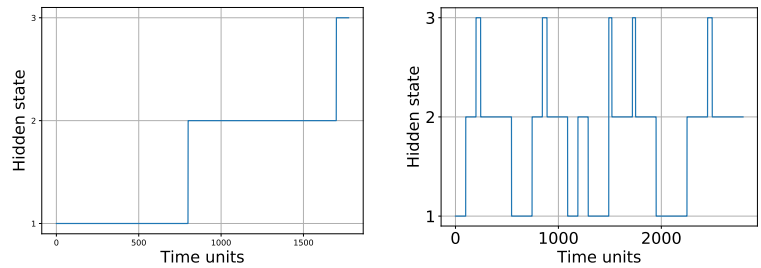
Two scenarios are considered with three known hidden states. One follows AR-AsLG-HMM emission probabilities and another AR-MoG-HMM emission probabilities. Blocks of data for each hidden state are generated. These blocks are mixed as indicated by Fig. 5.2 depending on the scenario to create a signal and train every model with it. The data is simulated as in real life applications where hidden states may not have a particular order of appearance i.e., any possible transition between hidden states is possible. The learned models are evaluated with two different types of sequences of hidden states. These two sequences are generated fifty times to be evaluated in the testing phase. From the fifty evaluations, the the mean log-likelihood (LL), mean BIC, the standard deviation of the log-likelihoods (Std) and the number of parameters in the model (#) are reported.

Both scenarios use six variables. From the parameters, in the case of AR-AsLG-HMM emission probabilities, a hidden state with no structural complexity is observed (no-AR and no-parent relationships in f_{im}^t), a second one with some structural complexity and the last one with a complex structure (several AR and parent relationships in f_{im}^t). The parameters in AR-AsLG-HMM are modified in such a way that the more complex the context-specific Bayesian networks are, the greater amplitudes for the $g_1(i)$ (Eq. (5.18)) function are. The parameters used for the hidden states for both AR-AsLG-HMM and AR-MoG-HMM can be



(a) (b)

Figure 5.2: Sequences of hidden states used to construct the training signals for scenario 1 (a) and for scenario 2 (b)



(a) Sequence 1 (b) Sequence 2

Figure 5.3: Sequences of hidden states used to construct the test signals. Sequence 1 (a) and Sequence 2 (b) are used for both scenarios

found in the Appendix A. The $g_1(i)$ function used for all the experiments has $v_m = 1$ for $m = 1, ..., M$ and $\kappa_m = 0$, for $m = 1, ..., M$. The sequence of hidden states used to construct the training signal for both scenarios can be seen in Fig. 5.2. The two sequences of hidden states used to generate the testing signals (fifty testing signals are generated for each sequence and each scenario) are illustrated in Fig. 5.3.

In Table 5.1 and Table 5.2, it can be observed the results for scenario 1 and 2 respectively, for both sequences. Note that AR-AsLG-HMM obtained the best results in LL and BIC score. The naïve-HMM, AsLG-HMM and AR-MoG-HMM obtained fair results. The mixture models: MoG-HMM, VAR-MVGHMM and BMM obtained poor results in LL and BIC score. In the case of BIC score, the penalization for mixture models was higher since a greater number of parameters were needed for these models. In terms of standard deviation, MoG-HMM and VAR-MVGHMM obtained the best results, nevertheless also they obtained the worst results in BIC score. Next, observe that AR-MoG-HMM, AR-AsLG-HMM and LMSAR obtained fair results in standard deviation with a good BIC score. Finally, AsLG-HMM and naïve-HMM obtained the worst standard deviation values in spite of their BIC score. In terms of the number of parameters, the naïve-HMM used the fewest number of parameters since it has the simplest structure and VAR-MVGHMM used the highest number of parameters since it considers cross-AR dependencies between variables. AR-AsLG-HMM and AsLG-

Seq	Model	mean LL	mean BIC	Std	#
1	AR-AsLG-HMM	**-25909.45**	**52432.48**	86.19	64
	AsLG-HMM	-32817.77	66181.78	193.74	55
	LMSAR	-30389.47	61587.06	112.36	108
	AR-MoG-HMM	-28960.25	59357.17	25.02	192
	MoG-HMM	-68411.13	138124.24	**1.67**	174
	Naïve-HMM	-33251.80	66997.46	199.35	**48**
	BMM	-56348.00	113997.98	19.58	174
	VAR-MVGHMM	-68243.34	140415.08	0.84	525
2	AR-AsLG-HMM	**-41478.86**	**83608.50**	87.76	64
	AsLG-HMM	-54167.33	108914.01	176.69	55
	LMSAR	-48356.20	97569.52	94.83	108
	AR-MoG-HMM	-45547.16	92618.09	32.21	192
	MoG-HMM	-107682.31	216745.54	**1.72**	174
	Naïve-HMM	-54395.72	109315.24	163.32	**48**
	BMM	-88693.42	178767.76	21.12	174
	VAR-MVGHMM	-107504.80	219176.15	1.50	525

Table 5.1: Results for each testing sequence of scenario 1

Seq	Model	mean LL	mean BIC	Std	#
1	AR-AsLG-HMM	**-19822.87**	**40573.45**	114.66	106
	AsLG-HMM	-20255.87	41304.78	118.92	88
	LMSAR	-26212.61	53233.23	112.16	108
	AR-MoG-HMM	-22990.46	47417.38	10.42	192
	MoG-HMM	-52213.44	105728.67	2.17	174
	Naïve-HMM	-23135.52	46764.83	138.46	**48**
	BMM	-40292.46	81886.72	23.05	174
	VAR-MVGHMM	-52069.19	108066.19	**0.76**	525
2	AR-AsLG-HMM	**-32883.36**	**66750.73**	192.35	106
	AsLG-HMM	-33867.52	68576.21	236.21	88
	LMSAR	-44482.64	89822.32	226.24	108
	AR-MoG-HMM	-36504.69	74533.01	14.41	192
	MoG-HMM	-82248.49	165877.78	3.95	174
	Naïve-HMM	-38804.18	78132.11	241.70	**48**
	BMM	-63655.48	128691.74	39.42	174
	VAR-MVGHMM	-82064.02	168294.22	**1.04**	525

Table 5.2: Scores for each testing sequence of scenario 2

HMM used fewer parameters than mixture models, whereas mixture models used between four to eleven times the number of parameters used by naïve-HMM. Note that LMSAR had a fair number of parameters since it assumes independence between variables for all hidden states, standard deviations independent of the hidden state and only AR parameters are used. In the Appendix A an analysis of the obtained Viterbi paths for some sequences can be found.

Model	Times 1 (s)	Times 2 (s)
AR-AsLG-HMM	6.842	55.098
AsLG-HMM	4.608	33.009
LMSAR	70.797	2.458
AR-MoG-HMM	189.114	223.762
MoG-HMM	110.749	190.766
Naïve-HMM	3.702	8.904
BMM	266.231	5762.679
VAR-MVGHMM	11.165	131.059

Table 5.3: Scenario 1 and 2 learning times

For training the models, the maximum number of EM iterations are set to 200 and the convergence threshold to 1×10^{-10}. For the SEM, the number of iterations are set to 200 and the convergence threshold to 1×10^{-5}. Table 5.3 shows the required times for learning the models in each scenario. All the models converged with some exceptions. In scenario 1, MoG-HMM and VAR-MVGHMM had to limit their number of EM iterations since singular-covariance matrices raised in the parameters, in particular, MoG-HMM had to iterate 8 times and VAR-MVGHMM iterated 26 times before singular covariance matrices appeared. In the case of scenario 2, LMSAR and MoG-HMM had similar problems and MoG-HMM iterated 13 times and LMSAR just could iterate 1 time. Note that BMM is the most expensive in time among all the models. This is due to the structure learning process that it does Bilmes [2003], where several mutual information quantities must be computed to determine the best AR-relationships. On the other hand, the fastest algorithm that converged was the naïve-HMM, which was expected since it had the simplest structure of all the models. In spite of that, observe that AR-AsLG-HMM and AsLG-HMM obtained the second best times for training, and the remaining models had longer training times.

From these experiments it can be concluded that AR-AsLG-HMM is capable of being simple enough to explain linear Gaussian autoregressive and mixture Gaussian processes and prevent overfitting, but can be complex enough to detect relevant parameters that drive the hidden states. AsLG-HMM has this property as well, but as can be seen from the obtained BIC scores and standard deviations, the AR variables are pertinent. In terms of variance of the predictions, AR-AsLG-HMM had decent results, which implies it is stable.

5.8.2 Air quality in Beijing

Here a dataset found in the UCI Machine Learning Repository named: "Beijing Multi-Site Air-Quality Data Data Set" Zhang et al. [2017] is used. The dataset consists of measurements of air quality in different monitoring stations in Beijing. In particular, it is taken the measurements from the file "PRSA Data Aotizhongxin" which represent the name of the monitoring station Aotizhongxin. This dataset has hourly measurements from March 2013 until February 2017. The data contains missing data (3.37% of the dataset for the selected variables). The missing data is filled using the mean of the values of the five previous hours. The hidden variable in this problem can be understood as the air quality. For this study, the following variables are used: sulfur dioxide (SO_2 in $\mu g/m^3$), nitrogen dioxide (NO_2 in $\mu g/m^3$), carbon monoxide (CO in $\mu g/m^3$), ozone (O_3 in $\mu g/m^3$), coarse particulate matter (PM_{10} in $\mu g/m^3$) and fine particulate matter ($PM_{2.5}$ in $\mu g/m^3$). Bayesian networks and HMM have been used before to determine air quality Vitolo et al. [2018], Vairo et al. [2019], Yang et al. [2017], Sun et al. [2013], showing advantages in the generation of information and discovery of relationships between variables.

The Chinese air quality limits for hourly, daily and monthly measurements are expressed in the law GB 3095-2012. These limits are used to model the g function for this problem. In particular, $\boldsymbol{\kappa} = \{500, 200, 10000, 200, 150, 75\}$ and $\mathbf{v} = \{1/500, 1/200, 1/10000, 1/200, 1/150, 1/75\}$. The g function in this case uses Eq. (5.19). If $g_2(i) > 0$ it means that one or many variables

Model	mean LL	mean BIC	Std	#
AR-AsLG-HMM	-167390.66	335516.71	1646.68	69
AsLG-HMM	-221071.33	442778.17	3183.77	58
LMSAR	-183257.57	366841.98	3034.64	36
AR-MoG-HMM	**-138174.57**	**277493.09**	**405.75**	126
MoG-HMM	-213999.24	429033.48	2692.62	114
Naïve-HMM	-228682.23	457745.78	2126.03	**30**
BMM	-214429.71	429894.41	2521.67	114
VAR-MVGHMM	-162813.92	326826.25	1393.19	132

Table 5.4: Air quality scores when two hidden states are used

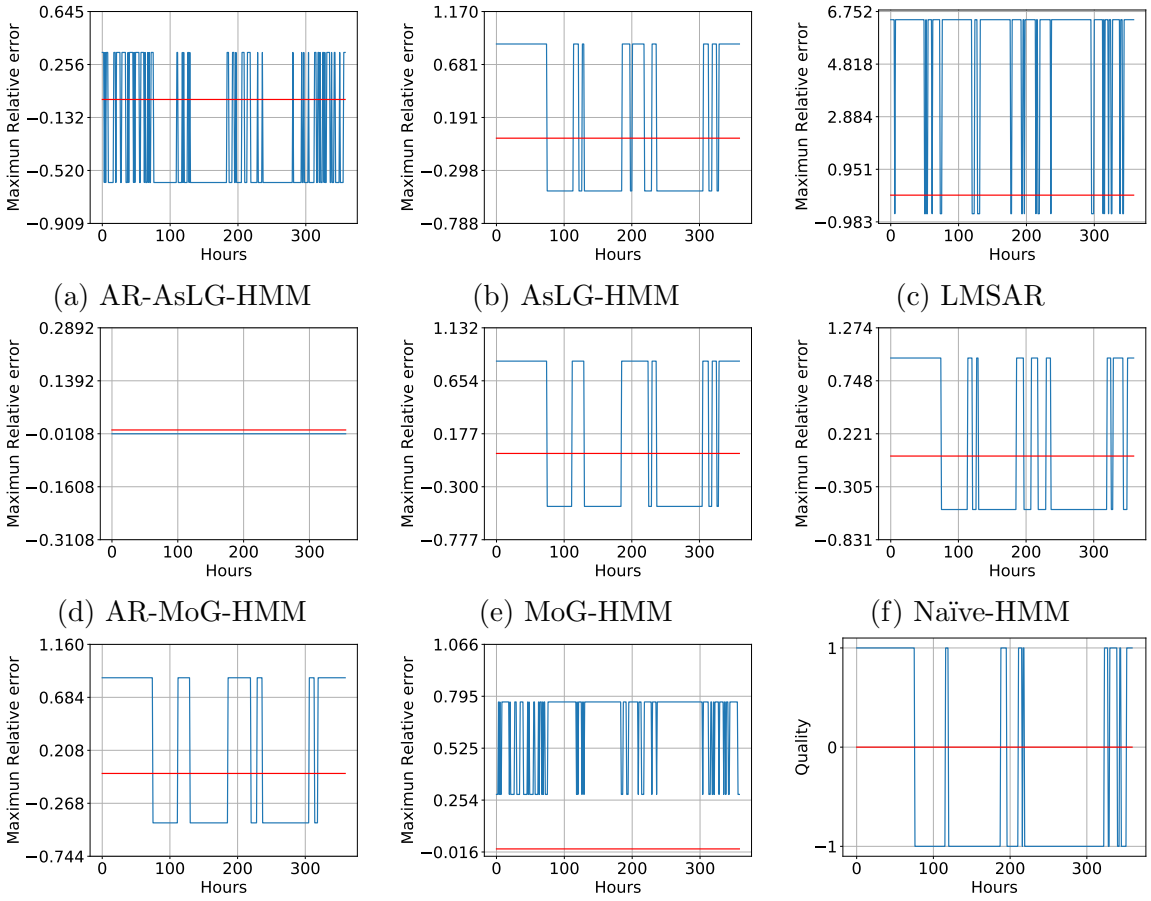
Model	mean LL	mean BIC	Std	#
AR-AsLG-HMM	-161077.29	323525.48	3346.94	133
AsLG-HMM	-214970.27	430930.13	3666.78	91
LMSAR	-186458.75	373898.02	3702.09	108
AR-MoG-HMM	**-138001.78**	**277746.70**	**210.11**	192
MoG-HMM	-211794.34	425168.40	5376.60	174
Naïve-HMM	-219751.80	440102.81	2570.88	**48**
BMM	-211524.89	424629.51	4456.12	174
VAR-MVGHMM	-155614.62	315995.65	2875.60	525

Table 5.5: Air quality scores when three hidden states are used

are above the permissible limit and the air quality is pretty bad. Great negative values are desirable for g_2 since it implies good air quality. The aim is to learn models to determine the air quality when new observations arrive. The first year of data is used to train the models: from march of 2013 to February of 2014. A first experiment with only two hidden states is considered with $p^* = 1$. This model is used to check if the model is capable of determining in a binary manner the air quality using AR processes of order one. Later, another model is trained where the number of hidden states is set using the naïve-HMM since this model is the simplest one. The selection of the number of hidden states could be done with the same AR-AsLG-HMM but for fairness, this strategy is used. Two to eleven hidden states were considered, but with three hidden states, naïve-HMM obtained the best LL for the year 2013 and all the models could be trained. For each model it is predicted individually the air quality of the three following years of data: from March of 2014 to February of 2017. As above, the mean likelihoods, mean BICs, standard deviation of likelihoods and number of parameters of each model are reported.

Table 5.4 and Table 5.5 show the scores obtained. Observe that AR-AsLG-HMM, VAR-MVGHMM and AR-MoG-HMM attained the best results in the LL and BIC scores. The remaining models got fair results. In terms of stability, AR-MoG-HMM has the lowest standard deviation, followed by BMM, naïve-HMM and AR-AsLG-HMM. In terms of the number of parameters, naïve-HMM and AsLG-HMM have the fewest number of parameters. Followed by these models, LMSAR and AR-AsLG-HMM achieved a fair number of parameters and finally, mixture models, as expected, had to use a great amount of parameters.

Fig. 5.4 shows the predicted air quality for the first two weeks of 2016 for each model using the Viterbi algorithm when two hidden states are used. Real readings are shown in Fig. 5.4(i), where 1 expresses when any of the variables surpasses the law limits and -1 when all the variables are under the law limits. From Fig. 5.4(i), there are four periods of time

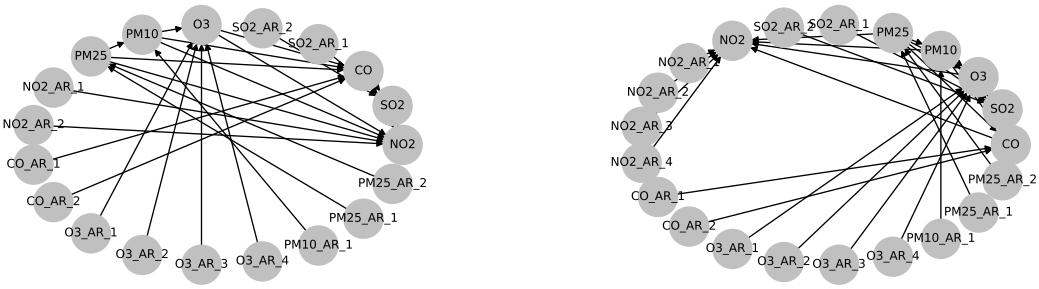


(a) AR-AsLG-HMM (b) AsLG-HMM (c) LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM (i) Real

Figure 5.4: Viterbi paths for the air quality example during the first week of 2016 when two hidden states are used



(a) $g(i) = -0.78$ (b) $g(i) = 0.7$

Figure 5.5: Context-specific graphs learned by AR-AsLG-HMM. (a) shows a graph where the air quality is good and (b), where the air quality is bad.

where pollution levels out of the legal levels are found: from 0 to 75 hours, from 115 to 120

hours, from 186 to 219 hours and from 322 to 360 hours. Clearly Fig. 5.4(i) does not tell us the severity of the pollution nor the closeness to an outlaw pollution level. The model with the highest score (AR-MoG-HMM Fig. 5.4(d)) shows a horizontal line below zero, which implies that the pollution level is always close to an outlaw level, which is not consistent with what is shown in Fig. 5.4(i). The next model with the highest LL and BIC is VAR-MVGHMM Fig. 5.4(h), which shows a noisy behaviour but always above zero, indicating a persistent outlaw pollution level with changes in severity; however it does not match with the reality observed in Fig. 5.4(i). In the case of LMSAR, there are transitions between legal and illegal pollution levels; however, it reads as persistent high pollution levels which are not consistent with Fig. 5.4(i). AR-AsLG-HMM shows a noisy prediction; however, in this case there are variations between outlaw levels and legal levels of pollution. There are four moments where the pollution levels are illegal in Fig. 5.4(a) as in Fig. 5.4(i); however, the prediction is not so clear as in the case of AsLG-HMM Fig. 5.4(b), MoG-HMM Fig. 5.4(e), naïve-HMM Fig. 5.4(f) and BMM Fig. 5.4(g), where more consistent predictions are found with similar levels of the $g_2(i)$ function. Since the Viterbi paths achieved for three hidden states are similar to those with two hidden states, they are shown in the Appendix A.

The noisy predictions can be explained using the learned transition matrices: in the case of non-AR models, the transition probabilities were concentrated on the diagonal of the transition matrix as in the case of AsLG-HMM in Eq. (5.21) with matrix $\boldsymbol{A_1}$; whereas AR models learned more uniform transition matrices as in the case of AR-AsLG-HMM in Eq. (5.21) with matrix $\boldsymbol{A_2}$.

$$\boldsymbol{A_1} = \begin{bmatrix} 0.96 & 0.04 \\ 0.04 & 0.96 \end{bmatrix}, \boldsymbol{A_2} = \begin{bmatrix} 0.80 & 0.20 \\ 0.38 & 0.62 \end{bmatrix} \tag{5.21}$$

The latter causes more likely jumps between hidden states and noisy Viterbi paths can be obtained. Nonetheless, from all the AR models, AR-AsLG-HMM was the only one closest to the real scenario given by Fig. 5.4(i).

Fig. 5.5 shows two learned graphs when three hidden states were assumed. In the context-specific Bayesian networks, AR variables are denoted as X_m_AR_r, where r is the number of lags for the variable X_m. In Fig. 5.5(a) it is shown a graph when the air quality is good and in Fig. 5.5(b) it is bad. In both graphs some interesting relationships similar to the ones found in Vitolo et al. [2018] are found. For example, in Fig. 5.5(a) CO depends on $PM_{2.5}$ and PM_{10} and SO_2 and NO_2 are related to CO. These relationships come from the process of combustion of gas and charcoal. Also NO_2 is related to O_3 which indicates the photochemistry of NO_2 for the production of O_3. In Fig. 5.5(b) these relationships remain. However, the dependences on previous values for each variable changes, which tells us the level of impact of the past on the pollution levels.

5.8.3 Ball bearings degradation

Ball bearings are used inside many mechanic tools as drills, rotors, etc. Ball bearings represent critical components inside these machines. The failure or degradation of these components can be translated to loses in time, money and assets for industries as explained in Chapter 1. Monitoring ball bearings is crucial and relevant, and the use of HMM can give insight of the bearing degradation process and therefore help in the development of maintenance policies Larrañaga et al. [2018].

The benchmark used to validate the proposed model in this section comes from the same ball bearing vibrational data from Section 4.3.1. In this case, the number of hidden states was set depending on the scores obtained by naïve-HMM in the training data. Observe that with seven hidden states, the scores of the naïve-HMM were optimized.

For this dataset, constantly it was obtained underflow problems due to the small amplitudes of some frequencies. Therefore, for this case study, all the dataset were multiplied by 1000. The g function uses $\boldsymbol{\kappa} = \{0, 0, 0, 0\}$ and $\mathbf{v} = \{1/1000, 1/1000, 1/1000, 1/1000\}$. Therefore if Eq. (5.18) is used, g_1 adds the magnitude of all the relevant frequencies. If there is a degradation in any of the ball bearing components, the relevant frequencies will have greater magnitudes and this will be perceived by g_1. Therefore, predicting the hidden state in the testing data can be seen as an approximation of the degradation of the ball bearing. The idea here is to train models such that they can determine the degradation state of forthcoming ball bearings. This can be accomplished with the Viterbi paths. In particular for this dataset the ball bearing 3 is interesting in S1 and S3, since it fails in both signals. Nevertheless, a model from S1 for all the ball bearings will be trained and it will be shown the scores obtained in the testing signal S3. Additionally, the Viterbi paths of ball bearing 3 will be pictured to see the respective degradation.

During the training time, the iterations of LMSAR and BMM had to be tuned to prevent problems with the variances or covariance matrices. Additionally, for the BMM, no structural optimization was performed, since it was unfeasible in time.

Table 5.6 shows the results obtained by the models for each ball bearing. Note that the best results in BIC were achieved by different models, say: AR-AsLG-HMM, LMSAR and VAR-MVGHMM. The worst results were attained generally by naïve-HMM and BMM. it is observed as well that MoG-HMM and AsLG-HMM got fair results but always worse than their AR counterparts (AR-MoG-HMM and AR-AsLG-HMM, respectively). In particular, in the case of B3, observe that the use of AR parameters improved significantly the LL and BIC scores. In terms of the number of parameters, see that naïve-HMM, AsLG-HMM and AR-AsLG-HMM used the least amount of parameters for all the ball bearings. The remaining models used two or three times the amount of parameters used by naïve-HMM. This implies that AR-AsLG-HMM fulfils its purpose of being a model which uses a reasonable amount of parameters with a good fit for new data.

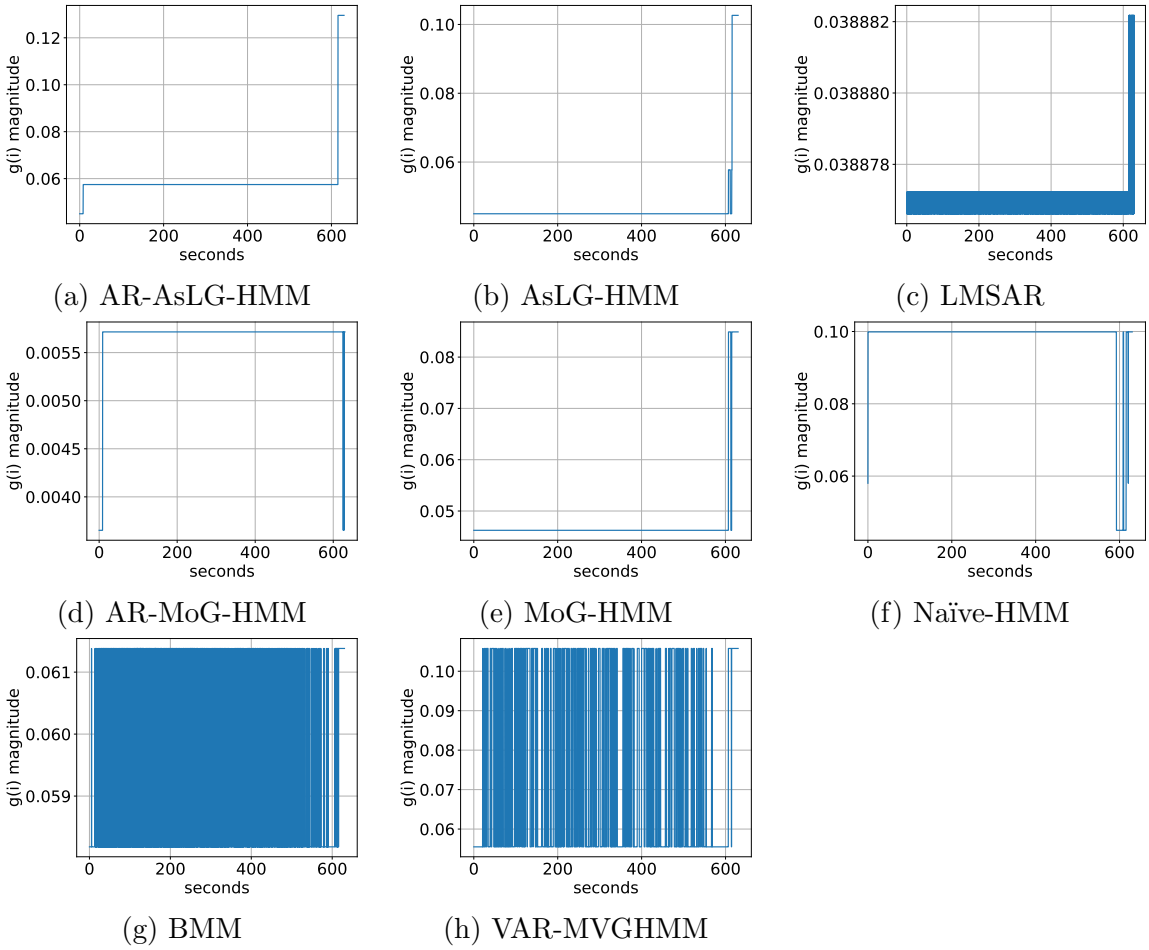
Fig. 5.6 shows the paths for the testing B3. it can be observed that AR-AsLG-HMM, AsLG-HMM, LMSAR and MoG-HMM exhibit the expected behaviour of the bearings degradation, since they maintain low $g_1(i)$ values during most of the bearing signal and $g_1(i)$ grows

B	Model	LL	BIC	#
B1	AR-AsLG-HMM	**-32095.06**	**64793.79**	57
	AsLG-HMM	-33086.47	66662.88	44
	LMSAR	-43531.13	87727.18	76
	AR-MoG-HMM	-37134.66	75424.18	132
	MoG-HMM	-42949.20	86738.30	96
	Naïve-HMM	-47619.04	95658.03	**36**
	BMM	-41397.04	83633.98	96
	VAR-MVGHMM	-35093.25	72338.73	246
B2	AR-AsLG-HMM	-38443.65	77438.49	51
	AsLG-HMM	-39527.63	79571.46	47
	LMSAR	**-29191.95**	**59048.81**	76
	AR-MoG-HMM	-31924.73	65004.32	132
	MoG-HMM	-37927.80	76695.49	96
	Naïve-HMM	-37296.65	75013.25	**36**
	BMM	-38705.65	78251.19	96
	VAR-MVGHMM	-35674.99	73502.21	246
B3	AR-AsLG-HMM	-56375.55	113424.77	65
	AsLG-HMM	-103975.42	208510.78	52
	LMSAR	**-44120.05**	**88800.04**	64
	AR-MoG-HMM	-44835.74	90616.38	108
	MoG-HMM	-107638.87	216117.64	96
	Naïve-HMM	-119225.39	238870.73	**36**
	BMM	-154597.02	310033.95	96
	VAR-MVGHMM	-108390.68	218513.65	198
B4	AR-AsLG-HMM	-32480.33	65748.06	78
	AsLG-HMM	-43628.14	87833.7	54
	LMSAR	-40498.35	81661.61	76
	AR-MoG-HMM	-38785.03	78724.91	132
	MoG-HMM	-42247.63	85335.15	96
	Naïve-HMM	-49034.67	98489.29	**36**
	BMM	-42300.06	85440.01	96
	VAR-MVGHMM	**-31443.86**	**65039.97**	246

Table 5.6: Model scores for ball bearing data

abruptly at the end of the bearing life. The models AR-MoG-HMM, Naive-HMM, BMM and VAR-MVGHMM show pure noise or non consistent Viterbi paths, i.e., the $g_1(i)$ function shows high values at the middle of the bearing signal and reduces its values at the end of the bearing life. In the case of LMSAR, the desired behaviour is observed but the differences in the $g_1(i)$ function between the end of the bearing life and the rest of the bearing signal are not significant which affects the model predictive power.

A relevant part of the proposed model is the generation of context-specific Bayesian networks. In Fig. 5.7 it is observed two context-specific Bayesian networks. Fig. 5.7(a) represents a good health state. In this graph observe that the cage frequencies (FTF) determine the remaining variables. This implies that knowing the behaviour of the cage, determines the behaviour of the ball bearing rollers and races. Fig. 5.7(b) represents a bad health state and shows a more complex structure. In this context-specific Bayesian network AR values are relevant and are taken into consideration. it can be noted again see the dominance of the ball bearings cage (FTF) to determine the dynamical process of the model, but some frequencies are not directly dependent on this variable e.g., the outer race frequencies (BPFO) depend on the inner race frequencies (BPFI) and the roller frequencies (BSF) and these depend directly on the cage frequency (FTF). In summary, these graphs are capable of explaining the ball bearings dynamical process depending on its health.



(a) AR-AsLG-HMM (b) AsLG-HMM (c) LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM

Figure 5.6: Sequence of hidden states by each model for B3

5.9 Conclusions

In this chapter, the development of asymmetric HMMs is extended to allow us to determine and learn the optimal number of time lags depending on the value of the hidden state via the SEM algorithm. Also, a greedy-forward heuristic is introduced to find the best structure for the model. Additionally, it has been theoretically adapted the forward-backward, Viterbi and EM algorithms to the proposed log-likelihood function. Furthermore, it was shown that every iteration of the EM algorithm improves the log-likelihood of the model.

A numerical labelling function is introduced, which can be helpful in determining the nature of the learned HMMs and to identify changes in the magnitude of the hidden variable.

Synthetic and real data were used to validate the proposed model, which was compared with many other models. In general, the AR-AsLG-HMM obtained good results in scores and predictions for synthetic and real data. it was showed the use of the learned context-specific Bayesian networks to extract information about the nature of the problem being modeled which is harder to obtain from traditional HMMs. Additionally, the number of parameters learned by AR-AsLG-HMMs were usually in an intermediate point between the simplest

(a) $g(i) = 0.03$ (b) $g(i) = 0.13$

Figure 5.7: Context-specific graphs learned by AR-AsLG-HMM. (a) shows a graph where the bearings health is good and (b), where the bearings health is bad.

model (naïve-HMM) and the mixture models, which is helpful to prevent data overfitting.

In the following chapters, this model will be used for offline/online FSS models and online health estimation of ball bearings, in order to meet the objectives of this thesis.

Chapter 6

Feature saliencies in AR-AsLG-HMMs

FSS has become an essential tool for data scientists to find relevant information within large datasets and reduce data dimensionality. However, little effort has been put into the case where no class variable is present. In this manner, many real unsupervised problems are neglected and many clustering models are forced to work with undesirable or irrelevant features. Recently, feature saliency (FS) models (Law et al. [2004]) have been proposed as an option to overcome the FSS problem with unlabeled data. A review of such models was presented in Section 3.2.3. HMMs with FS models are found in previous works ()Adams et al. [2016], Zheng et al. [2018], Adams and Beling [2020]). However, the models assume that all the variables are independent, which depletes their explanatory power. Consequently, in this chapter an FS model based on asymmetric HMMs is introduced to alleviate this issue, i.e., an HMM which is capable of simultaneously determining feature relevancy and giving a probabilistic dependency graph containing only the relevant features. A learning method is proposed based on the SEM algorithm. The proposed model is capable of performing inference in testing data. Synthetic and real data, coming from ball bearings and grammar facial videos, are used to validate the model. However, the model assumes some hypotheses such as: (i.) Variables must follow a linear Gaussian distribution; (ii.) the hidden states follow the Markov property; (iii.) the noise variables are Gaussian. Any dynamic process which drifts abruptly from these conditions is not suitable to be interpreted with the proposed model, since the data insights that the model would provide would be wrong.

The chapter organization is as follows: Section 6.1 introduces the new model based on AR-AsLG-HMMs with feature saliencies. Section 6.2 provides EM and SEM algorithms to update the parameters of the new feature saliency model. Section 6.3 provides a bound in big O notation of the computational complexity of an iteration of the learning process. Section 6.4 shows the results of applying the model to synthetic and real data. Section 6.5 gives the corresponding conclusions and comments related to the advantages and drawbacks of applying the new model.

6.1 Model proposal

In this contribution, assume that the emission probabilities are a mixture of Gaussian noise and autoregressive asymmetric linear Gaussian Bayesian networks. Thus, depending on the hidden state, the Bayesian network which describes the relevant distribution may change. This model will be referred to as FS-AsHMM.

The embedded FSS process assumes that irrelevant features are not affected by changes in hidden states, therefore a Bernoulli vector $\boldsymbol{Z}^t = (Z_1^t, ..., Z_M^t)$ is introduced in the model and the irrelevant behavior is modeled for each variable with a Gaussian distribution with parameters ϵ_m and τ_m^2. The dependency of $\boldsymbol{X}^t$ given $\boldsymbol{Z}^t$ and its at most p^* past values is modeled as:

$$\begin{aligned} b_i(\boldsymbol{x}^t|\boldsymbol{z}^t) &:= P(\boldsymbol{x}^t|\boldsymbol{x}^{t-p^*:t-1}, \boldsymbol{z}^t, Q^t = i, \boldsymbol{\lambda}) \\ &= \prod_{m=1}^{M} f_{im}(x_m^t)^{z_m^t} g_m(x_m^t)^{(1-z_m^t)}, \end{aligned} \tag{6.1}$$

where $f_{im}(x_m^t) = \mathcal{N}(x_m^t|\boldsymbol{u}_{im}^t\boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t\boldsymbol{\eta}_{im}, \sigma_{im}^2)$ is the relevant component, and $g_m(x_m^t) = \mathcal{N}(x_m^t|\epsilon_m, \tau_m^2)$ is the noise term, $\boldsymbol{u}_{im}^t = (1, u_{im1}^t, ..., u_{imk_{im}}^t)$ and $\boldsymbol{d}_{im}^t = (x_m^{t-1}, ..., x_m^{t-p_{im}})$ are vectors with the values of the k_{im} parents of X_m^t in the Bayesian network graph and its $p_{im} \leq p^*$ past values, with p^* an AR order fixed upper-bound. To be clear, the mean of the relevant term is the linear combination of the parameters $\boldsymbol{\beta}_{im} = (\beta_{im0}, \beta_{im1}, ...\beta_{imk_{im}})^\top$ and $\boldsymbol{\eta}_{im} = (\eta_{im1}, ...\eta_{imp_{im}})^\top$ with $\boldsymbol{u}_{im}^t$ and $\boldsymbol{d}_{im}^t$ respectively, and its variance is σ_{im}^2. The noise term for each variable X_m is a Gaussian distribution with mean ϵ_m and variance τ_m^2 which does not depend on the hidden state.

Observe in Fig. 6.1 an example of the new model topology. In this example, a network with two variables/features is presented. When $Q^t = 1$, no probabilistic relationships appear between X_1^t and X_2^t, also X_2^t depends on one AR value or X_2^{t-1}. When $Q^t = 2$, there is a probabilistic dependency of X_2^t from X_1^t, additionally, X_1^t depends on one AR value, that is X_1^{t-1} and X_2^t depends on two AR values or X_1^{t-1} and X_1^{t-2}. Finally, $\boldsymbol{X}^t$ on both contexts, $Q^t = 1$ and $Q^t = 2$, depends on the binary random vector $\boldsymbol{Z}^t$.

The probability of $\boldsymbol{Z}^t$ can be expressed as:

$$\zeta(\boldsymbol{z}^t) := P(\boldsymbol{z}^t|\boldsymbol{\lambda}) = \prod_{m=1}^{M} \rho_m^{z_m^t}(1-\rho_m)^{(1-z_m^t)} \tag{6.2}$$

$\rho_m := P(Z_m^t = 1|\boldsymbol{\lambda})$ for $m = 1, ..., M$. Note that it is assumed that the Z_m^t Bernoulli variables are independent between them and that the ρ_m parameters do not change with time. From

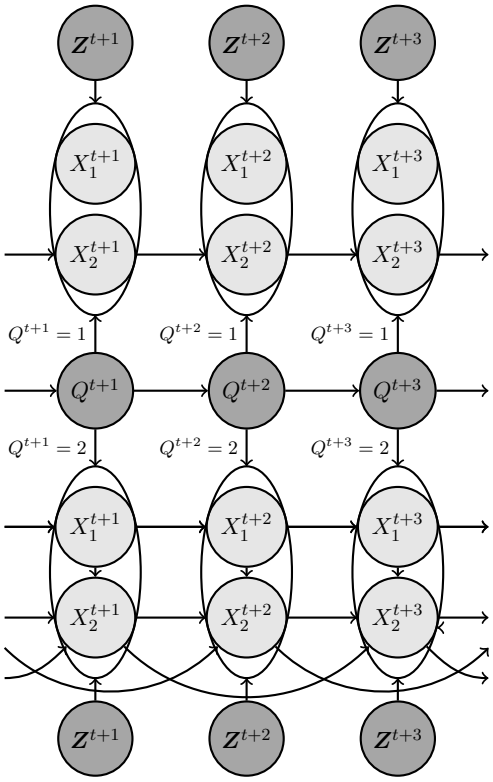


Figure 6.1: Example of a structure of a global feature saliency asymmetric HMM (FS-AsHMM)

Eq. (6.1) and Eq. (6.2) the emission probabilities can be derived:

$$\begin{aligned} b_i(\boldsymbol{x}^t) &:= P(\boldsymbol{x}^t|\boldsymbol{x}^{t-p^*:t-1}, Q^t = i, \boldsymbol{\lambda}) \\ &= \sum_{R(\boldsymbol{Z}^t)} P(\boldsymbol{x}^t, \boldsymbol{z}^t|\boldsymbol{x}^{t-p^*:t-1}, Q^t = i, \boldsymbol{\lambda}) \\ &= \sum_{R(\boldsymbol{Z}^t)} b_i(\boldsymbol{x}^t|\boldsymbol{z}^t)\zeta(\boldsymbol{z}^t) \\ &= \prod_{m=1}^{M} \rho_m f_{im}(x_m^t) + (1-\rho_m) g_m(x_m^t), \end{aligned} \tag{6.3}$$

and the full information probability can be written as follows:

$$P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) = \pi_{q^{p^*}} \prod_{t=p^*}^{T-1} a_{q^t q^{t+1}} \prod_{t=p^*}^{T} \zeta(\boldsymbol{z}^t) b_{q^t}(\boldsymbol{x}^t|\boldsymbol{z}^t). \tag{6.4}$$

6.2 Model learning

6.2.1 E-step

The auxiliary function is defined as:

$$\mathcal{Q}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) := \sum_{R(\boldsymbol{Q}^{p^*:T})} \sum_{R(\boldsymbol{Z}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \tag{6.5}$$

From Eq. (6.5), the log-likelihood (LL) of the $\boldsymbol{\lambda}$ model is:

$$\begin{aligned} \mathcal{Q}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) &= \mathcal{H}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + \ln P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \\ &= \mathcal{H}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + LL(\boldsymbol{\lambda}), \end{aligned} \tag{6.6}$$

where

$$\mathcal{H}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{R(\boldsymbol{Q}^{p^*:T})} \sum_{R(\boldsymbol{Z}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}). \tag{6.7}$$

By Eq. (6.6), Eq. (6.7) and the results in Section 5.2, it is known that each iteration of the EM algorithm with $\mathcal{Q}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ implies improvements in the modified likelihood function proposed in Eq. (5.1). Introducing Eq. (6.4) in Eq. (6.5), a tractable expression of $\mathcal{Q}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ is found, which will be useful to find the updating formulas of the model parameters:

$$\begin{aligned} \mathcal{Q}^{fs}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) &= \sum_{i=1}^{N} \gamma^{p^*}(i) \ln(\pi_i^{p^*}) + \sum_{t=p^*}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \xi^t(i,j) \ln(a_{ij}) \\ &+ \sum_{t=p^*}^{T} \sum_{i=1}^{N} \sum_{m=1}^{M} \psi_m^t(i) \ln(\rho_m f_{im}(x_m^t)) + \sum_{t=p^*}^{T} \sum_{i=1}^{N} \sum_{m=1}^{M} \phi_m^t(i) \ln((1-\rho_m) g_m(x_m^t)). \end{aligned} \tag{6.8}$$

In Eq. (6.8), the latent a posteriori probabilities are:

$$\begin{aligned} \gamma^t(i) &:= P(Q^t = i|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}), \\ \xi^t(i,j) &:= P(Q^{t+1} = j, Q^t = i|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}), \\ \psi_m^t(i) &:= P(Q^t = i, Z_m^t = 1|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}), \\ \phi_m^t(i) &:= P(Q^t = i, Z_m^t = 0|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}), \end{aligned} \tag{6.9}$$

for $t = p^*, ..., T$, $i = 1, ..., N$ and $m = 1, ..., M$. The E-step consists of estimating these quantities. In the case of $\psi_m^t(i)$:

$$\begin{aligned}\psi_m^t(i) &= P(Q^t = i, Z_m^t = 1|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \\ &= P(Z_m^t = 1|Q^t = i, \boldsymbol{x}_m^{t-p^*:t}, \boldsymbol{\lambda}^{(s)})\gamma^t(i) \\ &= \frac{\rho_m f_{im}(x_m^t)\gamma^t(i)}{\rho_m f_{im}(x_m^t) + (1-\rho_m)g_m(x_m^t)}.\end{aligned} \tag{6.10}$$

It is not hard to note that $\gamma^t(i) = \phi_m^t(i) + \psi_m^t(i)$ for $m = 1, ..., M$ and $i = 1, ..., N$. Therefore $\phi_m^t(i) = \gamma^t(i) - \psi_m^t(i)$ and:

$$\phi_m^t(i) = \frac{(1-\rho_m)g_m(x_m^t)\gamma^t(i)}{\rho_m f_{im}(x_m^t) + (1-\rho_m)g_m(x_m^t)}. \tag{6.11}$$

Then, $\gamma^t(i)$ can be computed as:

$$\gamma^t(i) = \frac{\alpha_{p^*}^t(i)\beta_{p^*}^t(i)}{\sum_{j=1}^N \alpha_{p^*}^t(j)\beta_{p^*}^t(j)}. \tag{6.12}$$

In the previous equation the forward variable is $\alpha_{p^*}^t(i) := P(Q^t = i, \boldsymbol{x}^{p^*:t}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}^{(s)})$ and the backward variable is $\beta_{p^*}^t(i) := P(\boldsymbol{x}^{t+1:T}|Q^t = i, \boldsymbol{x}^{0:t}, \boldsymbol{\lambda}^{(s)})$. The forward-backward algorithm stated in Section 5.3 must be applied to estimate $\alpha_{p^*}^t(i)$ and $\beta_{p^*}^t(i)$. Finally, $\xi^t(i,j)$ can be computed as:

$$\xi^t(i,j) = \frac{\alpha_{p^*}^t(i)a_{ij}b_j(\boldsymbol{x}^{t+1})\beta_{p^*}^{t+1}(j)}{\sum_{u=1}^N\sum_{v=1}^N \alpha_{p^*}^t(u)a_{uv}b_v(\boldsymbol{x}^{t+1})\beta_{p^*}^{t+1}(v)}. \tag{6.13}$$

6.2.2 M-step

The M-step corresponds to optimizing Eq. (6.8) with respect to the model parameters. The following theorem gives the updating formulas that result from the optimization.

Theorem 6.1. Assume there is a current model $\boldsymbol{\lambda}^{(s)}$ such that the E-step has been computed with it. From optimizing Eq. (6.8), the resulting parameter $\boldsymbol{\lambda}^{(s+1)}$ can be obtained with the following updating formulas.

The feature saliencies $\{\rho_m^{(s+1)}\}_{m=1}^M$ are updated as:

$$\rho_m^{(s+1)} = \frac{\sum_{i=1}^N\sum_{t=p^*}^T \psi_m^t(i)}{T+1-p^*}. \tag{6.14}$$

The initial distribution $\boldsymbol{\pi}^{(s+1)} = \{\pi_i^{(s+1)}\}_{i=0}^N$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{6.15}$$

The transition matrix $\mathbf{A}^{(s+1)} = \{a_{ij}^{(s+1)}\}_{i,j=1}^{N}$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{6.16}$$

The mean and variance, $\{\epsilon_m^{(s+1)}\}_{m=1}^{M}$ and $\{(\tau_m^2)^{(s+1)}\}_{m=1}^{M}$, from the noise component, are updated as:

$$\begin{aligned} \epsilon_m^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i) x_m^t}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)} \\ (\tau_m^2)^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)(x_m^t - \epsilon_m)^2}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)}. \end{aligned} \tag{6.17}$$

Setting $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ for $m = 1, ..., M$, $t = p^*, ..., T$ and hidden state $i = 1, ..., N$, the parameters $\{\eta_{imr}^{(s+1)}\}_{r=1}^{p_{im}}$ and $\{\beta_{imk}^{(s+1)}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear system:

$$\begin{cases} \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t = \sum_{t=p^*}^{T} \psi_m^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{6.18}$$

if $\boldsymbol{\theta}_{im} = (\boldsymbol{\beta}_{im} | \boldsymbol{\eta}_{im})^\top$, $\boldsymbol{o}_{im}^t = (\boldsymbol{u}_{im}^t | \boldsymbol{d}_{im}^t)$, and $\boldsymbol{\Psi}_{im}^{p^*:T} := \text{Matrix}([\psi_m^{p^*}(i), ..., \psi_m^T(i)])$. Then, the previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{6.19}$$

Setting $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$; then, $\{(\sigma_{im}^2)^{(s+1)}\}_{i,m=1}^{N,M}$ can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_m^t(i)(x_m^t - \hat{\varphi}_{im}^t)^2}{\sum_{t=p^*}^{T} \psi_m^t(i)}. \tag{6.20}$$

The proof of this theorem is provided in the Appendix B. It is worth noting that, from Eq. (6.18), for each variable $m = 1, ..., M$ and hidden state $i = 1, ..., N$, the size of the linear system will depend on the number of parents and AR values; the longer the list of dependencies, the larger the linear system.

6.2.3 The SEM algorithm

In this chapter, the greedy-forward algorithm proposed in Section 5.6 is used to search the space of possible graphical models. However, in this model, it is plausible to think that if a variable is noise, it should not be considered in any explanatory graphical model. Therefore, a restriction during the search of structures is imposed, such that no noise variable is added to any context-specific Bayesian network. The restriction consists of omitting any possible arc coming to or from variables X_m which fulfills the following condition: $\rho_m \leq \bar{\rho}$, where $\bar{\rho} \in [0, 1)$ is a threshold that determines which variables are relevant. Observe that the opposite of this assumption is not true, i.e., if a variable does not have any relationship with any other variable in a context-specific Bayesian network, it does not mean that it is irrelevant or noise under our relevance definition.

6.3 Computational complexity

Routine	Complexity
Means	$O(NMT(M+p^*))$
Probabilities	$O(TMN)$
Forward-Backward	$O(TN^3)$
Viterbi	$O(NT(M(M+p^*)+N))$
E-step	$O(TN(N^2+M))$
M-step	$O(NM(M+p^*)^2(M+p^*+T))$
Graph scoring	$O(NM(M+p^*)((M+p^*)^2+T))$

Table 6.1: Computational complexity of different routines of the learning and inference algorithms

In Table 6.1, the computational complexity in big O notation is shown of the different routines of the proposed algorithm. It is assumed that the learned networks are dense or that several arcs appear in the context-specific Bayesian networks and that $p^* << T$, or that the maximum lag of the AR processes is small compared to the length of the data.

Given a prior or current model $\boldsymbol{\lambda}$, the *Means* and *Probabilities* routines refer to computing and storing the temporal means ν_{im}^t and probabilities $\{b_i(\boldsymbol{x}^t)\}_{i=1}^N$, $\{f_{im}(x_m^t)\}_{i=1,m=1}^{N,M}$, $\{g_m(x_m^t)\}_{m=1}^M$ for $t = p^*, ..., T$, which are required to perform the *Forward-Backward*, *Viterbi* and *E-step* routines. The *Forward-Backward* routine refers to the computation of the forward variable $\{\alpha_{p^*}^t(i)\}_{i=1}^N$ and backward variable $\{\beta_{p^*}^t(i)\}_{i=1}^N$, $t = p^*, ..., T$ in Eq. (6.12). These can be used to compute log-likelihoods or perform the *E-step* in the EM algorithm. The *E-step* consists of computing the latent probabilities in Eq. (6.9) and the *M-step* is to update the parameters of $\boldsymbol{\lambda}$ using Theorem 1. Then, the *Means* and *Probabilities* routines are again executed if another EM iteration is needed. Finally, the *Graph-scoring* routine refers to the evaluation of a new set of graphs or context-specific Bayesian networks during the SEM algorithm. It means using the *Means*, *Probabilities* routines and Eq. (6.18) and Eq. (6.20).

6.4 Experimental setup

In this section, the proposed model will be compared with the models in Adams et al. [2016] (FS-HMM) and Zheng et al. [2018] (SHMM-LFS). Since these models have been previously compared in favor to clustering FS models such as Zhu et al. [2012], Li et al. [2009] and Nguyen et al. [2015], they are omitted in the experiments. Additionally, the model AR-AsLG-HMM will be compared to observe the advantages and disadvantages of using feature saliency models. Synthetic data and real data from face grammatical videos and degradation datasets of ball bearings are used for validation. In the case of Zheng et al. [2018], the number of mixture components is fixed to 1 for the synthetic data; since the data is structured to behave like that. In the case of real data, 2 components were only used since additional components drastically increased the number of parameters to be estimated, as will be seen later. Finally, for all the experiments, $\bar{\rho} = 0.9$. Recall that this value determines which variables cannot be in the context-specific Bayesian networks. Further details are given in the following subsections. For the sake of space, simply AsHMM will mean AR-AsLG-HMM.

6.4.1 Synthetic data

6.4.1.1 Data description

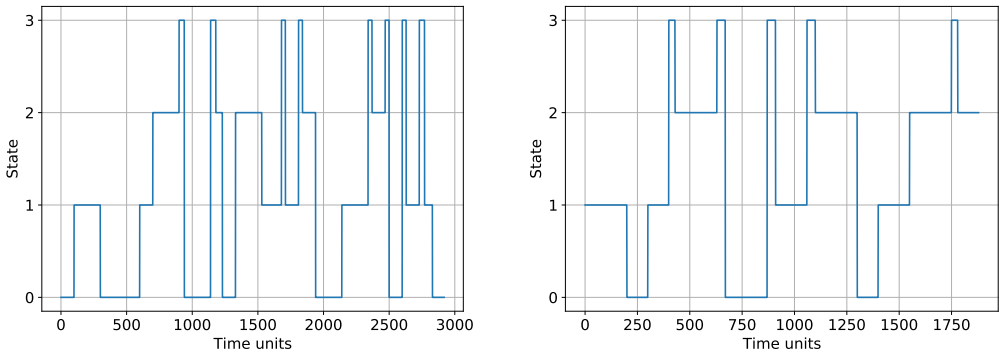
In this study, a synthetic data set with no physical interpretation is built. It contains noise variables, partial noise variables and relevant variables. The noise variables are those which follow a normal distribution with fixed mean and variance. Partial noise variables are those whose parameters do not change for every drift in the hidden state in the full data. Relevant variables follow a normal distribution whose mean and variance change with every drift in the hidden state in the data set. Additionally, it is assumed that noise variables have no probabilistic relationship with other variables. The data is built with probabilistic relationships between variables and different AR values and is assumed to have four hidden states and ten variables.

Three possible scenarios are analyzed, i.e., three sets of parameters are used. The set of parameters are described in the Appendix B. In all the scenarios, the variables with index 3 and 10 are considered as noise. In scenarios 1 and 2, the variable with index 5 is also noise. In scenario 3, the variables with index 5 and 7 are considered as partial noise variables. Now, with respect to the dependency maps of each scenario: The first scenario assumes that all the variables are independent. The second scenario assumes that probabilistic relationships between relevant variables may appear. The third scenario is the most complex, due to the presence of probabilistic relationships between variables and AR values, and some variables are partial noise. This information is summarized in Table 6.2.

The training data is generated following the sequence of hidden states exposed in Fig. 6.2 (a). For the testing phase, a testing sequence of hidden states is used, which is pictured in Fig. 6.2 (b). The testing sequence is generated fifty times for each scenario for population evaluation purposes. To evaluate and compare the models, the mean LL ($\bar{\text{LL}}$), BIC and the standard deviation of LL (σ_{LL}) are used.

S	# Dep	# AR	Noise	P. noise
1	0	0	X_3, X_5, X_{10}	-
2	28	0	X_3, X_5, X_{10}	-
3	17	17	X_3, X_{10}	X_5, X_7

Table 6.2: Synthetic data global description. S denotes the scenario case. # Dep stands for the number of dependencies between variables. # AR represents the number of AR dependencies. Noise denotes the noise variables. P. noise stands for the partial noise variables



(a) (b)

Figure 6.2: Signals used for training and testing. (a) is used for training and (b) is used for testing

6.4.1.2 Synthetic data results

S	Model	$\overline{\text{LL}}$	$\overline{\text{BIC}}$	σ_{LL}	#
1	AsHMM	**-30527.66**	**61824.36**	9.95	**102**
	FS-HMM	-30600.85	62181.84	9.93	130
	FS-AsHMM	-30537.76	62063.21	**9.91**	131
	SHMM-LFS	-48217.39	98696.64	18.02	300
2	AsHMM	**-30526.87**	**61815.24**	9.57	**101**
	FS-HMM	-31816.89	64613.93	9.68	130
	FS-AsHMM	-31706.78	64431.40	**9.52**	135
	SHMM-LFS	-46062.65	94387.16	16.24	300
3	AsHMM	**-33855.63**	**68955.54**	**0.52**	165
	FS-HMM	-43263.58	87507.51	17.57	**130**
	FS-AsHMM	-34531.54	70239.50	18.57	156
	SHMM-LFS	-702438.3	1407138.94	73.56	300

Table 6.3: Results for the test sequence for the different compared models

In Table 6.3 the obtained results for the different models in terms of LL and BIC are shown. Additionally, Fig. 6.3 shows the critical difference diagrams Demšar [2006] with a confidence of 90% for the obtained rankings in the testings datasets for LL and BIC scores, respectively. In this case the Nemenyi test was used, which evaluates for all pairs of models the hypothesis of no difference in the ranking position. The *CD* value indicates the minimum distance in the rank to give evidence of statistical difference. In the graphs, models grouped by the same bold-line are not statistically different in their rank. In our critical diagrams, 1 is the best rank and 4 is the worst rank.

From Fig. 6.3 it can be observed that the model with the best BIC in mean was AsHMM,

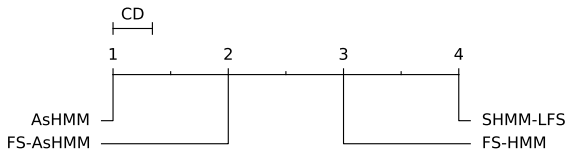


Figure 6.3: Critical difference diagram with the Nemenyi hypothesis test for the ranking of BICs

Model	Q	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5
		ρ_6	ρ_7	ρ_8	ρ_9	ρ_{10}
FS-HMM		0.98	1.0	0.07	0.81	0.07
		1.0	0.98	1.0	0.93	0.06
FS-AsHMM		0.98	0.99	0.08	0.82	0.1
		1.0	0.98	1.0	0.93	0.11
SHMM-LFS	1	1.0	1.0	1.0	1.0	1.0
		1.0	1.0	1.0	1.0	1.0
	2	1.0	1.0	1.0	1.0	1.0
		1.0	1.0	1.0	1.0	1.0
	3	1.0	1.0	1.0	1.0	1.0
		1.0	1.0	1.0	1.0	1.0
	4	1.0	1.0	1.0	1.0	1.0
		1.0	1.0	1.0	1.0	1.0

Table 6.4: ρ results in scenario 3 for the different models in the synthetic data

followed in order by FS-AsHMM, FS-HMM and SHMM-LFS. By the Nemenyi test, it can be noted that all models were statistically different since no bold line paired pairs of models. The same results were obtained for the rankings with LL.

In terms of standard deviations of the prediction scores, SHMM-LFS obtained the worst results in all the scenarios. Next, FS-AsHMM obtained the best standard deviation in scenario 1 and 2, but it increased in scenario 3 were AsHMM obtained the best result.

For the number of parameters (last column in Table 6.3), AsHMM obtained the least amount in scenarios 1 and 2 in spite of the fact that FS-HMM assumed full independence between variables. However, in scenario 3, FS-HMM used the least number of parameters since AsHMM increased drastically its number of parameters. It must be noted that in all the scenarios, the number of parameters of the asymmetric models changed; this is due to the different context-specific Bayesian networks that were found during the learning phase. It is also noticeable that the SHMM-LFS model obtained the largest amount of parameters in spite of the fact that this model also assumed full independence between variables. This model in particular, in contradiction with the definition of irrelevant features Law et al. [2004], assumes that the noise distribution also changes its parameters with the hidden state and mixture component, which increases its number of parameters drastically.

It has been shown that AsHMM obtained good fitness and does not require a critical amount of parameters. Nonetheless, the proposed models introduce feature saliencies which are capable of estimating feature relevancy and provide more data insights. Following this idea, in Table 6.4, the estimated relevancies by each model for scenario 3 are shown. Recall from the synthetic data description that variables 3 and 10 are represented as noise, i.e., their parameters do not change with the hidden states; and variables 5 and 7 are represented

as partial noise variables, i.e., their parameters do not change for all the hidden states. Observe that FS-HMM and FS-AsHMM were capable of detecting the noise or irrelevant variables since the relevancies of ρ_3 and ρ_{10} are close to zero; while relevancies of partial noise variables, ρ_5 and ρ_7, obtained mixed results. The remaining values have higher relevancy but with contrasts, e.g., the relevancies ρ_2, ρ_6 and ρ_8 are close to one ($\rho > 0.9$) but for relevancies ρ_4, it is not clear if they are totally relevant or not ($0.5 < \rho < 0.9$).

In the case of SHMM-LFS, the relevancies change with the hidden states. In Table 6.4 the column Q refers to each (numbered) hidden state. Regarding SHMM-LFS, the feature saliencies are estimated at the component level in a mixture of Gaussians; however, as previously mentioned, for comparative purposes, mixture models with only one component are considered in the synthetic data. In this case, observe that for SHMM-LFS all the features are predicted as relevant, for all the hidden states and features.

Model	t-S1 (s)	t-S2 (s)	t-S3 (s)
AsHMM	25.97	20.05	26.23
FS-HMM	53.99	144.30	22.79
FS-AsHMM	39.08	169.53	61.54
SHMM-LFS	**14.61**	**14.64**	**14.21**

Table 6.5: Seconds to learn a model by scenario in the synthetic data

Concerning the execution times of the tested algorithms, in Table 6.5, it is reported the learning times of all the algorithms for each scenario. Note that, FS-AsHMM was the largest time consumer in two out of the three scenarios. In scenario 1, FS-HMM was the slowest model to learn. In this manner, it is observed that the additional information that was provided by FS-AsHMM and FS-HMM (feature saliencies) had the cost of longer training times. SHMM-LFS was the quickest in all the scenarios. However, as seen before, the data insights obtained by this algorithm were poor. In an intermediate point, AsHMM can be found. The algorithm is capable of giving context-specific Bayesian networks, but no feature selection is performed.

6.4.2 Grammatical facial expression data

6.4.2.1 Data description

In this experiment, grammatical facial expression data is used. Facial gestures are relevant in any sign language: given a hand signal sequence, facial gestures can change its grammatical sense. The data by Freitas et al. [2014] was collected using a Kinect camera which recorded fluent Brazilian sign language signalers. From the videos, spacial facial points were collected, processed and used for classification and segmentation tasks. In each video, there was a person repeating, with pauses, sentences with grammar content. The aim of the problem is to determine in which frames the grammatical content is being performed. HMMs have been used before to tackle this kind of problem. See for example Michael et al. [2009], Nguyen and Ranganath [2012]. The possible grammar contents are:

- **Affirmative** (*Affir*): used to make positive sentences;
- **Conditional** (*Cond*): used to create subjunctive clauses;
- **Doubt question** (*DQ*): used to indicate that new information is being added;
- **Emphasis** (*Emp*): used to highlight information;
- **Negative** (*Neg*): used to make negative sentences;
- **Relative sentence** (*Rela*): used to provide more information;
- **Topic** (*Topic*): used to change subject;
- **WH-questions** (*WH*): used to create who, what, where... questions;
- **Y/N-questions** (*YN*): used to create yes/no questions.

Later, another expert in Brazilian sign language labeled the video frames indicating when the signaler is performing the sentence. Each video has a unique grammar content, where five repetitions of five different sentences in which the face is not overshadowed by the hand signals are recorded. Each dataset has spatial (x, y, z) coordinate information from 100 facial points (300 raw features). However, by expert knowledge, the dataset can be reduced down to 18 features which contain information about distances and angles between relevant face sections in the (x, y) plane Freitas et al. [2014]. More details about the raw features can be found in Fig. 6.4 and Table 6.6.

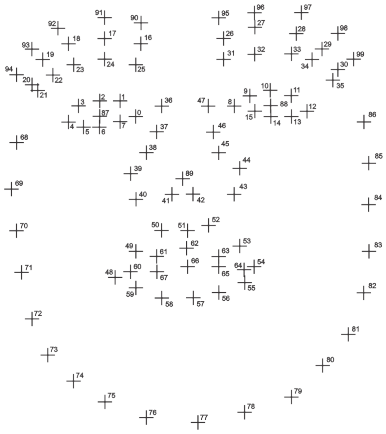


Figure 6.4: Raw face point locations

The idea is to select relevant points from relevant face parts. In Table. 6.7 the 18 extracted features are described, namely $d_1, ..., d_{11}$, which are distances between face points, and $a_1, ..., a_7$, which are angles between face points, denoted as $||\boldsymbol{a}-\boldsymbol{b}|| = \sqrt{(a[x]-b[x])^2 + (a[y]-b[y])^2}$ and $\angle(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) = \arccos \frac{(\boldsymbol{b}-\boldsymbol{a})\cdot(\boldsymbol{c}-\boldsymbol{a})}{||\boldsymbol{b}-\boldsymbol{a}||||\boldsymbol{c}-\boldsymbol{a}||}$. Two people perform the same sentences for each grammatical context. Three of the five repetitions of the two signallers are taken as input for a model for each grammar content. Later, the models are evaluated with the remaining two

Point	Face section	Selected points
$\mathbf{0-7}$	Left eye	2
$\mathbf{8-15}$	Right eye	10
$\mathbf{16-25}$	Left eyebrow	17
$\mathbf{26-35}$	Right eyebrow	27
$\mathbf{36-47}$	Nose	39, 44
$\mathbf{48-67}$	Mouth	48, 51, 54, 57
$\mathbf{68-86}$	Face contour	-
$\mathbf{87}$	Left iris	-
$\mathbf{88}$	Right iris	-
$\mathbf{89}$	Nose tip	89
$\mathbf{90-94}$	Line above left eyebrow	-
$\mathbf{95-99}$	Line above right eyebrow	-

Table 6.6: Raw selected points

Expert knowledge reduced dataset			
d_1	d_2	d_3	d_4
$\|\|\mathbf{2-17}\|\|$	$\|\|\mathbf{17-27}\|\|$	$\|\|\mathbf{27-10}\|\|$	$\|\|\mathbf{10-89}\|\|$
d_5	d_6	d_7	d_8
$\|\|\mathbf{89-2}\|\|$	$\|\|\mathbf{39-89}\|\|$	$\|\|\mathbf{89-44}\|\|$	$\|\|\mathbf{44-57}\|\|$
d_9	d_{10}	d_{11}	a_1
$\|\|\mathbf{57-39}\|\|$	$\|\|\mathbf{57-51}\|\|$	$\|\|\mathbf{48-54}\|\|$	$\angle(\mathbf{10,17,2})$
a_2	a_3	a_4	a_5
$\angle(\mathbf{2,27,10})$	$\angle(\mathbf{89,48,54})$	$\angle(\mathbf{48,89,51})$	$\angle(\mathbf{54,81,51})$
a_6	a_7		
$\angle(\mathbf{48,51,57})$	$\angle(\mathbf{54,51,57})$		

Table 6.7: Features construction

repetitions of both signallers for each grammar content (18 sequence tests). The mean $\overline{\text{LL}}$ and $\overline{\text{BIC}}$ scores are reported (standard deviation is not reported, since only two test sequences are available for each grammar content). From the supervised binary problem (grammar content or not), the accuracy, recall and F-score from the classification task are provided.

6.4.2.2 Grammatical face data results

From the previous information, the models BICs and LLs for each grammar case are reported. As the aim of this problem is to obtain a binary segmentation of the recorded videos, two hidden states will be used. The accuracy, recall and F-score obtained by each model will also be reported. Next, the model corresponding to the *Topic* grammar case is explored. From it, its feature saliencies and learned Bayesian networks are presented and analyzed. In this manner, the additional data insights that the proposed models can provide are highlighted.

In Table 6.8 the LLs, BIC scores and number of parameters obtained by the different models for each grammatical scenario are shown. Additionally, Fig. 6.5(a) shows the critical difference diagrams for the BIC score. A confidence level of 90% was used. The critical difference diagram from LL score was omitted since it was the same as in the case of BIC.

Note that, AsHMM obtained the best BIC score and LL; followed in order by LFS-AsHMM, FS-HMM and SHMM-LFS. From the hypothesis test it can be observed that FS-AsHMM was statistically better than SHMM-LFS; but not enough evidence is available to confirm that it performed better than FS-HMM nor worse than AsHMM.

In Fig. 6.5(b) the critical difference diagram for the number of parameters are drawn. FS-

Grammar	Model	$\overline{\text{LL}}$	$\overline{\text{BIC}}$	#
Afir	AsHMM	**-42403.96**	**85935.22**	183
	FS-HMM	-51617.50	104048.14	**132**
	FS-AsHMM	-51182.32	103220.91	139
	SHMM-LFS	-351868.77	706879.23	510
Cond	AsHMM	**-76062.93**	**154090.58**	289
	FS-HMM	-97076.42	195050.22	**132**
	FS-AsHMM	-93579.59	188178.92	150
	SHMM-LFS	-670067.66	1343602.47	510
DQ	AsHMM	-66560.91	133622.67	**78**
	FS-HMM	-65544.13	131935.84	132
	FS-AsHMM	**-63677.90**	**128267.58**	142
	SHMM-LFS	-458428.4	920131.53	510
Emp	AsHMM	**-40022.70**	**81753.47**	275
	FS-HMM	-53869.89	108559.64	**132**
	FS-AsHMM	-53864.29	108548.44	**132**
	SHMM-LFS	-368888.71	740945.10	510
Neg	AsHMM	**-59059.31**	**120344.83**	349
	FS-HMM	-74477.48	149796.97	**132**
	FS-AsHMM	-75139.11	151196.76	144
	SHMM-LFS	-453618.42	910490.05	510
Rela	AsHMM	**-77468.23**	**157457.23**	363
	FS-HMM	-109543.42	220003.48	**132**
	FS-AsHMM	-107576.03	216110.36	138
	SHMM-LFS	-774777.30	1553096.18	510
Topic	AsHMM	**-69567.09**	**140984.06**	277
	FS-HMM	-85695.22	172271.97	**132**
	FS-AsHMM	-82010.55	165036.20	152
	SHMM-LFS	-588667.95	1180741.82	510
WH	AsHMM	**-35070.70**	**72076.86**	318
	FS-HMM	-47792.39	96388.17	**132**
	FS-AsHMM	-47804.90	96413.2	**132**
	SHMM-LFS	-327078.86	657261.75	510
YN	AsHMM	**-52210.02**	**106155.85**	270
	FS-HMM	-68438.54	137725.68	**132**
	FS-AsHMM	-67808.19	136477.85	134
	SHMM-LFS	-465458.38	934195.49	510

Table 6.8: Likelihood, BIC score and number of parameters obtained by the models for the testing face grammar videos

AsHMM was better ranked (less parameters) than SHMM-LFS, but not significantly different from FS-HMM and AsHMM. As a final comment, it is remarkable that SHMM-LFS obtained the highest amount of parameters again, as in the synthetic data.

The proposed problem is to learn a model capable of predicting from a video whether or not a signaler is performing a certain grammatical face expression. Therefore, a model for each training set is learned and then the corresponding testing videos are evaluated. However, as disclaimed, all the models are unsupervised, and they do not take into account the class variable. It follows that the generated models may not be segmenting or clustering the actions corresponding to the class variable.

The prediction phase is performed using the Viterbi algorithm. The Viterbi algorithm in this case, as only two hidden states are considered, returns sequences of zeros and ones. However, it is not clear what a zero or a one implies in this sequence; therefore, the confusion matrix is computed for the two following possible assignments:

- 1 is a grammar expression, 0 is not

Grammar	Model	Acuracy	Recall	F-score
Afir	AsHMM	0.61	0.0	0.0
	FS-HMM	**0.70**	**0.79**	**0.67**
	FS-AsHMM	0.68	0.67	0.62
	SHMM-LFS	0.62	0.0	0.0
Cond	AsHMM	0.68	**0.85**	0.61
	FS-HMM	**0.81**	0.77	**0.70**
	FS-AsHMM	**0.81**	0.68	0.67
	SHMM-LFS	0.72	0.0	0.0
DQ	AsHMM	0.57	0.0	0.0
	FS-HMM	0.56	0.52	0.50
	FS-AsHMM	**0.59**	**0.53**	**0.53**
	SHMM-LFS	0.58	0.0	0.0
Emp	AsHMM	0.63	0.59	0.51
	FS-HMM	**0.89**	**0.91**	**0.84**
	FS-AsHMM	0.85	0.85	0.78
	SHMM-LFS	0.67	0.0	0.0
Neg	AsHMM	0.56	0.37	0.43
	FS-HMM	**0.58**	0.41	0.47
	FS-AsHMM	0.56	**0.49**	**0.50**
	SHMM-LFS	0.55	0.0	0.0
Rela	AsHMM	0.55	0.54	0.40
	FS-HMM	**0.86**	**0.95**	**0.79**
	FS-AsHMM	**0.86**	0.90	**0.79**
	SHMM-LFS	0.72	0.0	0.0
Topic	AsHMM	0.72	0.71	0.54
	FS-HMM	**0.86**	**0.91**	**0.75**
	FS-AsHMM	0.85	0.87	0.73
	SHMM-LFS	0.77	0.0	0.0
WH	AsHMM	0.56	0.36	0.42
	FS-HMM	0.75	0.54	0.65
	FS-AsHMM	**0.77**	**0.74**	**0.74**
	SHMM-LFS	0.56	0.0	0.0
YN	AsHMM	0.72	**0.77**	0.69
	FS-HMM	0.73	0.66	0.66
	FS-AsHMM	**0.80**	0.71	**0.74**
	SHMM-LFS	0.60	0.0	0.0

Table 6.9: Prediction scores obtained by the models for the testing face grammar videos

		Distance										
M	Q	ρ_{d_1}	ρ_{d_2}	ρ_{d_3}	ρ_{d_4}	ρ_{d_5}	ρ_{d_6}	ρ_{d_7}	ρ_{d_8}	ρ_{d_9}	$\rho_{d_{10}}$	$\rho_{d_{11}}$
1		0.47	0.44	0.9	0.93	0.09	0.5	0.56	0.26	0.5	0.91	0.98
2		0.29	0.31	1.0	0.96	0.02	0.44	0.66	0.11	0.22	0.99	0.96
3	1	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	2	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93

		Angle						
M	Q	ρ_{a_1}	ρ_{a_2}	ρ_{a_3}	ρ_{a_4}	ρ_{a_5}	ρ_{a_6}	ρ_{a_7}
1		0.27	0.33	0.42	0.81	0.85	0.27	0.23
2		0.07	0.12	0.09	0.84	0.79	0.04	0.03
3	1	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	2	0.93	0.93	0.93	0.93	0.93	0.93	0.93

Table 6.10: ρ relevancies for the case of the grammatical case of *Topic*. Column M refers to the model: 1 is FS-HMM model, 2 is FS-AsHMM and 3 is SHMM-LFS. Column Q refers to hidden states; it is only used when a model has relevancies that depend on the hidden state (models 3 and 4)

- 0 is a grammar expression, 1 is not

From the confusion matrix of each assignment, the accuracy is computed and the assignment with the greater accuracy is chosen as the model segmentation. Nevertheless, a better decision rule could have been made if an expert in Brazilian sign language had reviewed the learned

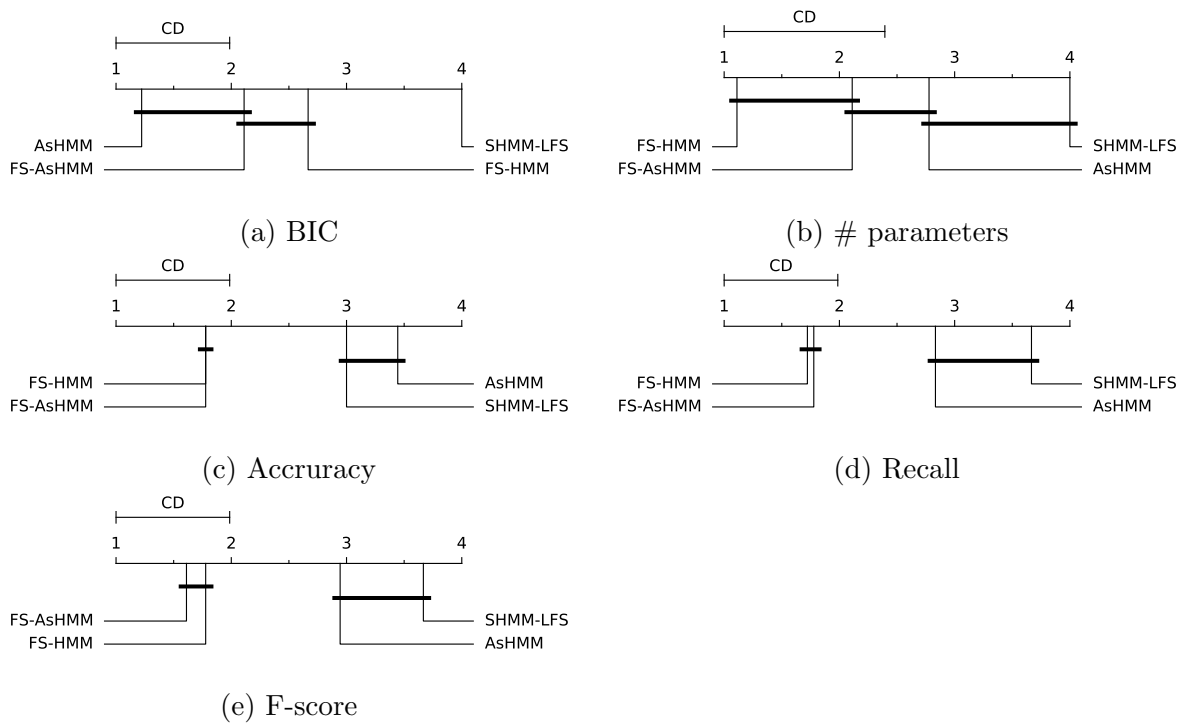


Figure 6.5: Critical difference diagram with the Nemenyi hypothesis test for the ranking of (a) BIC scores, (b) number of parameters, (c) Accuracy, (d) Recall and (e) F-score

parameters.

In Table 6.9 the total accuracy, recall and F-score obtained from the testing videos from signallers A and B are shown. Additionally, in Fig. 6.5(c), Fig. 6.5(d) and Fig. 6.5(e) the corresponding critical difference diagrams are provided. For accuracy (Fig. 6.5(c)), it can be observed that the best ranked models were FS-AsHMM and FS-HMM, followed in order by SHMM-LFS and AsHMM. Statistically, two equivalence groups were found, the first one consisted of FS-HMM and FS-AsHMM, and the second one contained AsHMM and SHMM-LFS.

In terms of recall or true positive rate, Fig. 6.5(d) shows that the best ranked model was FS-HMM, followed in order by FS-AsHMM, AsHMM and SHMM-LFS. In Table 6.9 it can be observed that there are models with high recall score and low accuracy , and models with zero recall but with a relevant accuracy. The main reason for this issue is that the output of the Viterbi algorithms for some models and grammar cases is a constant line at 0 or 1. Thus, this measure is not helpful to differentiate between the tested models. Therefore, the F-score can be used to solve this issue and give another perspective of the model performance for detecting grammar facial expressions.

Then, for the F-score, Fig. 6.5(e) shows that the best rank was obtained by FS-AsHMM, followed in order by FS-HMM, AsHMM and SHMM-LFS. FS-AsHMM is statistically better than AsHMM and SHMM-LFS but statistically equivalent to FS-HMM.

In conclusion, from the accuracy level and F-score, the most accurate and reliable models

were FS-AsHMM and FS-HMM for classification of this kind of data. AsHMM obtained intermediate or poor accuracy and F-score results.

It has been shown from the previous results that FS-AsHMM and FS-HMM can be good unsupervised models to predict grammar facial expressions. But it is also relevant to check their feature saliencies and observe which variables were relevant for the learning process. In Table 6.10, the relevancies for each model in the grammar case of *Topic* are presented. Note that, FS-HMM and FS-AsHMM (models 1 and 2) selected the same variables under the rule $\rho_i > \bar{\rho}$, i.e., the variables d_3, d_4, d_{10} and d_{11}. The remaining variables are not relevant or lie in an intermediate level. In any case, only the previous mentioned variables are considered in the context-specific Bayesian network inside FS-AsHMM and are able to have probabilistic relationships with other variables. Regarding SHMM-LFS (model 3), all the variables are relevant for both hidden states, which is undesirable since no FSS procedure is performed.

Observe that from FS-AsHMM and FS-HMM, the set of relevant features is small, only four from the eighteen variables are relevant. In this sense, it can be argued that AsHMM learned noise and predicted noise during the testing phase. In comparison, FS-AsHMM and FS-HMM also learned noise, but are capable of detecting the level of noise in the variables and use it during the determination of context-specific Bayesian networks and the prediction phase. This may explain why AsHMM obtained good scores in BIC and log-likelihood but performed poorly when the class variable was considered.

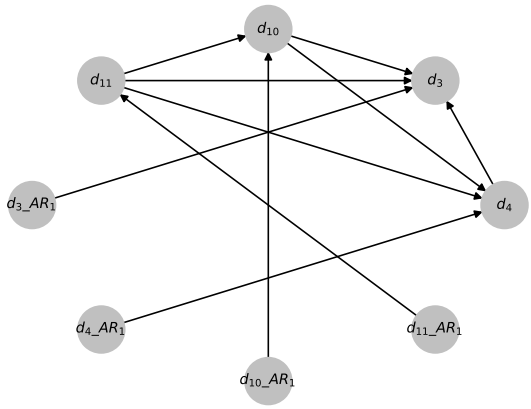


Figure 6.6: Learned context-specific Bayesian networks from the FS-AsHMM model for the facial grammar *Topic*

Finally, the learned context-specific Bayesian networks are interpreted. In Fig. 6.6, one of the networks learned from the FS-AsHMM model is pictured, for the *Topic* grammar when the grammar expression is being performed. In the graph, the nodes of the context-specific Bayesian network are labeled as *variable_AR_#order*; where *AR_#order* is the AR order. If it is 0, then this suffix is ignored for the label since it is the original variable. In this case, the

four relevant distances interact between them. In particular, d_{11} is the distance that governs the network since the remaining distances depend on this one. Also, it is remarkable that one AR value is relevant for all the selected distances, which indicates that the previous (in time) distances have an impact on the current ones. This kind of information can be useful for domain experts to determine or validate how the different face sections interact between them for the different grammar contents. Finally, note that these networks are only available for asymmetric models. Models such as FS-HMM and SHMM-LFS are not capable of giving this kind of insight.

Model	$\bar{t}/T$ (s)	σ_t/T (s)
AsHMM	0.0287	0.0073
FS-HMM	0.0126	0.0053
FS-AsHMM	0.0305	0.0277
SHMM-LFS	**0.0058**	**0.0004**

Table 6.11: Mean and standard deviation learning times per unit data for the grammatical facial expression dataset

Regarding the computational cost of the tested algorithms in the grammatical facial expression dataset, in Table 6.11 a brief summary of learning times is shown. As the dataset for learning each model has a different length, the time per unit of data is used to estimate the time performance of the four algorithms. Observe that, in mean and variance, the most expensive algorithm was FS-AsHMM, followed by AsHMM, FS-HMM and SHMM-LFS. In this dataset, the proposed algorithm was 2.42 times slower compared to its version without context-specific Bayesian networks (FS-HMM) and 1.06 times slower compared with its version without feature saliencies (AsHMM). Finally, in spite of the fast learning times of SHMM-LFS, the learned models were not capable of extracting relevant information from data.

6.4.3 Ball bearing degradation data

6.4.3.1 Data description

To validate the proposed models in the case of real ball bearings, the benchmark and data provided in Section 4.3.1 is used. In this work, the raw vibrational signals is filtered using spectral Kurtosis algorithms and extract relevant bearing fundamental frequency amplitudes such as the BPFO, BPFI, BSF and FTF. From these four fundamental frequencies, three harmonics are taken into account (a total of 16 variables). It is known that harmonic frequency magnitudes become more relevant when a failure is present Jáuregui-Correa and Lozano-Guzman [2020]. However, it is expected that some harmonic components can be more relevant than others. Nonetheless, the idea of this study is to determine the level of relevancy of the different harmonics when a failure is present. In the training dataset, signal 1 ball bearing 3 (S1 B3) fails due to its inner race and Bearing 4 (S1 B4) due to its rollers. In the testing dataset, signal 3 ball bearing 3 (S3 B3) fails due to its outer race. Bearing

3 is the most relevant mechanical component for this case studio, since it fails in both the training and testing dataset.

6.4.3.2 Ball bearing data results

In Table 6.12 the results obtained in the test by each model are shown. Observe that for all the cases, the model with the best fitness was AsHMM followed always in order by FS-AsHMM, FS-HMM and SHMM-LFS. For B1 and B2, it can be observed that for all the model, omitting SHMM-LFS, the maximum relative difference with respect to the best BIC score was of 1.85% . Meanwhile, in B3, the differences are larger; in the case of FS-AsHMM, the relative difference with respect to the best BIC score model was 10.28%, whereas FS-HMM obtained a much higher difference of 42.52%. This can be explained because B3 is the failing bearing. Due to its exponential behavior in the failing phase, AR parameters and probabilistic dependencies between features can play an important role in explaining this behavior Puerto-Santana et al. [2022b]; since FS-HMM assumes full independence, these dependencies were ignored. In the case of B4, only FS-AsHMM was close to the best scoring model with a relative difference of 3.93% and FS-HMM 16.47%.

B	Model	LL	BIC	#
B1	AsHMM	**-607437.06**	**1217153.25**	261
	FS-HMM	-619005.39	1239722.32	**196**
	FS-AsHMM	-618342.31	1238518.40	210
	SHMM-LFS	-4587984.01	9183966.81	916
B2	AsHMM	**-616773.52**	**1236088.15**	291
	FS-HMM	-618441.27	1238594.07	**196**
	FS-AsHMM	-618198.49	1238274.43	215
	SHMM-LFS	-4587984.01	9183966.81	916
B3	AsHMM	**-616748.69**	**1236710.86**	368
	FS-HMM	-880412.04	1762535.61	**196**
	FS-AsHMM	-680386.46	1363811.76	348
	SHMM-LFS	-4587984.01	9183966.81	916
B4	AsHMM	**-470522.75**	**945918.13**	558
	FS-HMM	-550028.31	1101768.14	**196**
	FS-AsHMM	-489337.82	983129.11	510
	SHMM-LFS	-4587984.01	9183966.81	916

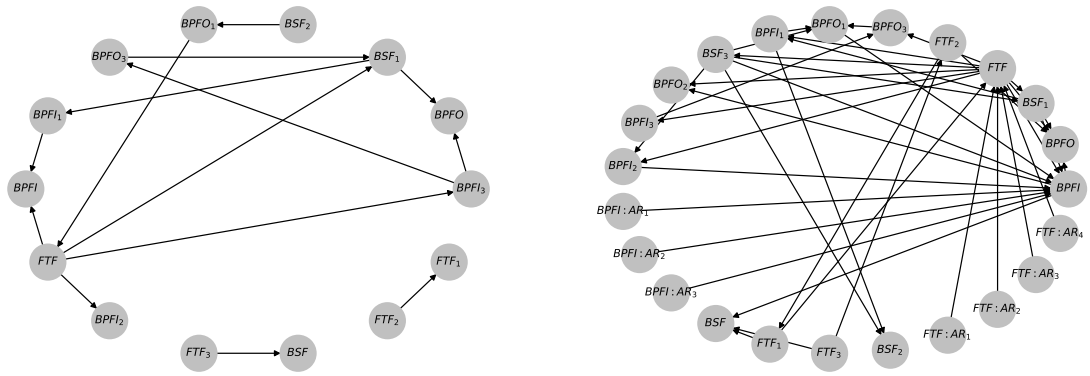
Table 6.12: LLs, BICs and number of parameters of the models in the testing signals for all the bearings B1, B2, B3 and B4

In terms of parameters, FS-HMM obtained the least amount of parameters in all cases. Since this model assumes full independence variables, no further parameters are added as opposed to the asymmetric model. On the other hand, for all the ball bearings SHMM-LFS uses the highest amount of parameters. Observe that, for each ball bearing, the number of parameters changes again for each asymmetric model in spite of the fact that all of them are of the same kind. In particular, notice that in ball bearings B1 and B2, the amount of parameters is much lower than in B3 and B4, this is because, in the training phase, B3 and B4 fail.

Although the proposed feature saliency models in this case study did not obtain the best results in BIC or LL when compared to AsHMM; note that the feature saliencies provided by the proposed models can give further data insights which AsHMM is not able to provide.

		BPFO				BPFI			
Model	Q	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	ρ_7	ρ_8
FS-HMM		0.83	0.85	0.81	0.80	0.88	0.83	0.83	0.77
FS-AsHMM		1.0	0.99	0.99	0.96	1.0	1.0	1.0	0.98
SHMM-LFS	1	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	2	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	3	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	4	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
		BSF				FTF			
M	Q	ρ_9	ρ_{10}	ρ_{11}	ρ_{12}	ρ_{13}	ρ_{14}	ρ_{15}	ρ_{16}
FS-HMM		0.80	0.82	0.8	0.86	0.90	0.82	0.82	0.78
FS-AsHMM		0.98	0.99	0.94	0.99	0.99	0.98	0.98	0.96
SHMM-LFS	1	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	2	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	3	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
	4	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93

Table 6.13: Learned feature saliencies for each model in the case of B3. Column Q refers to hidden state; it is only used when a model has relevancies that depend on the hidden state (models 3 and 4)



(a) (b)

Figure 6.7: Learned context-specific Bayesian structures from the FS-AsHMM model for the ball bearing B3. (a) corresponds to the graph learned in a low degradation state, whereas (b) corresponds to a high degraded state

From the dataset description, it is known that the training and testing ball bearing B3 fails. Therefore, more attention to the feature saliencies from B3 is given. In Table 6.13 the learned feature saliencies for each model are shown. The sixteen features are divided into four groups. Features with index $1, 5, 9$ and 13 correspond to the fundamental frequencies BPPFO, BPFI, BSF anf FTF respectively. The next three indices for each fundamental frequency correspond to its first, second and third harmonic. e.g., index $2, 3$ and 4 are the first, second and third BPFO harmonic.

From Table 6.13, observe that for FS-HMM the FTF fundamental frequency is the most relevant feature with a relevancy of 0.9. This result is unexpected since in the training phase, the whole degradation process is being observed and therefore more harmonics and fundamental frequencies should be relevant. Yet, it is remarkable that the relevancy of some fundamental frequencies, say BPFI and FTF, is higher than the relevancy of their harmonics. Meanwhile for BPFO, only the first harmonic has a greater relevancy than the fundamental

frequency; and in the case of BSF, the fundamental frequency has lower relevancy than its harmonics, being the third harmonic the frequency with the largest relevancy among the BSF frequencies.

In the case of FS-AsHMM, all the frequencies (fundamental and harmonics) are relevant since their relevancy fulfills the condition $\rho_i \geq \bar{\rho}$, this can be explained because all the degradation process of the ball bearing is being considered during the training phase. It is worth mentioning that for the BPFO and FTF frequencies, the larger the harmonic, the lower the relevancy. Also, as in the case of FS-HMM, BSF harmonics could obtain greater relevancies when compared to the relevancy of the fundamental frequency. If the relevancy threshold was tighter, for example $\overline{\rho} = 0.99$, the relevant frequencies would have changed. In this case the third harmonic of the BPFO and BPFI frequency, the fundamental and second harmonic of the BSF frequency and the first, second and third harmonic of the FTF frequency would be irrelevant. Additionally, for the FS-HMM model, no variable would be relevant.

As previously mentioned, the proposed models do not assume full probabilistic dependency or independence between variables as other models do. Fig. 6.7 shows a pair of context-specific Bayesian networks learned by the FS-AsHMM corresponding to low and high degradation levels.

In the case of the Bayesian network in Fig. 6.7 (a), it represents a low degradation level. Note that different probabilistic relationships appear. For instance, the FTF fundamental frequency has the most amount of descendants: the BSF first harmonic and the BPFI fundamental, second and third harmonic. Additionally, the BSF fundamental depends on the third FTF harmonic and the BPFO fundamental relies on the first BSF harmonic which also depends on the FTF fundamental. From this kind of information, it is plausible to say that in a low degradation state, for this bearing, the cage of the ball bearing is leading the dynamical system. Fig. 6.7 (b) shows another learned context-specific Bayesian network from a more degraded ball bearing state. In this case, more variables and dependencies appear in the graph, in particular, AR variables are considered. For example, three and four AR values of the BPFI and FTF fundamental frequencies respectively appear and can be helpful to describe late degradation stages of the ball bearing.

Model	t-B1 (s)	t-B2 (s)	t-B3 (s)	t-B4 (s)
AsHMM	551.57	127.48	144.31	220.75
FS-HMM	24.25	48.00	46.02	**15.57**
FS-AsHMM	27.64	57.28	108.91	78.19
SHMM-LFS	**16.29**	**16.49**	**15.85**	19.14

Table 6.14: Times to learn a model for each ball bearing in seconds

Regarding the learning times for this application, they are displayed in Table 6.14. In this case, the slowest algorithm was AsHMM, followed by FS-AsHMM, FS-HMM and SHMM-LFS. As in the previous datasets, SHMM-LFS was the fastest algorithm in most scenarios, but it captured little relevant information from data. On the other hand, note that FS-AsMM had big differences in the learning times when compared to FS-HMM. Recall that B3 and B4

broke and therefore its dynamic behavior is more complex compared to B1 and B2. This is evidenced by looking at Table 6.12 where B3 and B4 models for FS-AsHMM obtained more parameters than B1 and B2 FS-AsHMM models.

6.5 Conclusions

In this chapter, a new model based on AR-AsLG-HMM was introduced to perform offline FSS in an embedded manner. This new model is capable of estimating the relevant features and locally-optimal context-specific Bayesian networks for the selected relevant features, all during their learning phase. The parameter learning procedure for each model is detailed and proved. Also, a restriction to the space of context-specific Bayesian networks is imposed in order to consider only relevant features in the graph construction.

Experiments with synthetic and real data from grammar facial expressions and ball bearing wearing data are considered. For the experiments other two other state-of-the-art models were used for comparative purposes, namely FS-HMM (Adams et al. [2016]) and SHMM-LFS (Zheng et al. [2018]). For the latter model, a theoretic argument was given to validate its little usefulness to detect relevant features in the experiments. Additionally, these two models consider that all variables are independent, which in many real case scenarios is not true.

From the experiments, it was observed that the proposed model can obtain fair results in fitness. In the case of synthetic data, it was noted that it is capable of detecting irrelevant features. When evaluating grammar facial expressions, FS-AsHMMs and FS-HMMs obtained good results in accuracy and F-score, in spite of the fact that As-HMM obtained better results in fitness. Additionally, the proposed model was capable of determining which variables were noise and not useful for prediction.

From the algorithm complexity point of view, big O notation bounds were provided for the different routines that the algorithm can perform. The learning times for the proposed algorithm were among the highest for all the datasets. However, it was observed that the times were shorter when the data was simpler (see the ball bearing case). This property caused high variance in the obtained times which indicates that the model is capable of giving further data insights when needed at the cost of higher computational times; otherwise, the times are closer to those obtained by simpler models.

In conclusion, although models such as As-HMMs can obtain better results in BIC and log-likelihood than the feature saliency models, when noise variables are present, the learning and testing performance of As-HMM can be senseless: it is learning and predicting noise. When feature saliencies are added to the model, these were capable of discriminating between noise and relevant features. Additionally, it was proven that the proposed model was able to generate context-specific Bayesian networks at the same time as the feature saliencies were estimated. In this manner, the restriction of total independence in feature saliency models is overcome.

Chapter 7

AR-AsLG-HMM for online monitoring

In this chapter, the AR-AsLG-HMM model is adapted and applied to work online in data-streams for ball bearing surveillance. The goal of this chapter is to estimate in an online manner the health and ball bearing RUL. For the health estimation issue, a health index function (HI) is proposed (an *ad hoc* hidden state labeling function as in Section 5.7), such that it uses the model parameters to provide an estimation of the ball bearing degradation level. Such HI is later used for the RUL estimation. For the elaboration of this chapter, edge computing nodes were provided by Aingura IIoT, and code optimization techniques were provided by the Barcelona Computer Center to execute and embed AR-AsLG-HMMs into the edge device. Additionally, several issues regarding signal sampling, signal processing times and computational learning times had to be addressed in order to ensure data integrity. Therefore, computational complexity and memory management analysis were taken into consideration for this application. This chapter is a summary of the results obtained in Puerto-Santana et al. [2022a]. As reviewed in Section 3.3 and in Table 3.3, Puerto-Santana et al. [2022a] meets the five-point criteria that no other technique in the state of the art attains, which are:

(**1**.) It uses HMMs

(**2**.) It can be used in online environments

(**3**.) It does not require RTF data

(**4**.) It estimates the tool RUL

(**5**.) it self-updates when new trends in data appear

In this chapter, the techniques reviewed in Section. 2.4 are used to detect concept drifts in data. Real data is used from the FEMTO repository (Nectoux et al. [2012]) and a mechanical setup at Aingura IIoT for ball bearings RUL estimation.

The contents of this chapter are sorted as follows: Section 7.1 provides the steps and details to apply AR-AsLG-HMMs into online environments for ball bearing health and RUL estimation. Section 7.2 describes the real datasets used to validate the methodology. Section 7.3 shows the ball bearing health and RUL estimations from the datasets considered. Also, a memory consumption analysis of the algorithm inside the edge device is performed to detect the weak and strong points of the current methodology implementation. Finally, Section 7.4 provides the comments and conclusions to the proposed methodology.

7.1 Proposed methodology

In this section, no new model is introduced, but rather a set of steps to create an adaptive model which evolves over time when novel concepts appear. Fig. 7.1 shows a flow diagram indicating the proposed methodology workflow. In short, the methodology consists of repeating the following steps as required:

1. Capture new data and process it using signal processing algorithms.
2. Compute the processed data fitness with the current As-HMM.
3. Use the Page test and Chernoff bounds to detect new trends in data.
4. Update the model and add new hidden states to the As-HMM if needed.
5. Use the Viterbi algorithm with the current As-HMM and compute the HI.
6. Use the HI historic to estimate the RUL.

Observe that in the diagram in Fig. 7.1 there are three shaded boxes, namely *Novelty detection*, *Model update* and *RUL prediction*. The *Novelty detection* box includes the algorithms of BIC computation, the Page test (Section 7.1.2.1) and the Chernoff bounds (Section 7.1.2.2). The *Model update*, it consists of the model online training that is explained in Section 7.1.1. With respect to *RUL prediction*, a health index is proposed (Section 7.1.3.1) and a linear regression is used to estimate the RUL (Section 7.1.3.2). For a good performance of the proposed methodology, the following assumptions are taken:

1. The sensors are sensitive enough to capture the ball bearing degradation process to failure and not only the bearing failure phase.
2. No health recoveries are observed by the sensors.
3. No control affects the dynamical behavior during the data acquisition.

The first assumption refers to data quality. The second assumption, as can be seen below, ensures a proper RUL prediction, and the third assumption prevents the controlled behaviors from being considered as degradation.

As this methodology is expected to be used in online environments, emission probabilities such as mixtures or a multivariate normal Gaussian can compromise the model and data

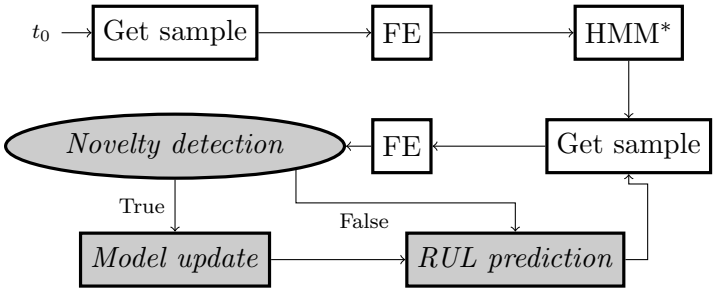


Figure 7.1: Flow diagram of the novelty detection and RUL prediction strategy. FE stands for feature extraction. HMM* stands for training a new HMM. *Novelty detection* looks for new trends in data. *Model update* updates the model and the baseline health. *RUL prediction* computes the proposed health index and estimates the ball bearing RUL

acquisition integrity since they are slower to compute due to the use of unnecessary parameters Puerto-Santana et al. [2022b]. Therefore, an AR-AsLG-HMM (which is called As-HMM for the sake of space) is used for this methodology, since it uses less parameters to model data and therefore, inference processes are more reliable and faster. In particular, the networks are selected by their fitness computed via the BIC score; whereas the search method is based on a greedy-forward explore scheme proposed in Section 5.6 which verifies that the networks are directed acyclic graphs (DAGs).

7.1.1 Model update

The proposed methodology uses an As-HMM which evolves when a concept drift is detected. For the initial HMM, the model is trained as explained in Section 5.2, with only one hidden state. However, for online data, the learning process must be different, and the SEM batch algorithm stated in Section 5.6, is manipulated to fit the proposed requirements. Additionally, the number of hidden states is not fixed a priori and can change with the data-stream.

Assume that the novelty detection methodology (which will be explain in Section 7.1.2) detects a novel concept at time $t + L$ with $L \in \mathbb{N}$ and the current model is no longer valid to explain the chunk of data $\boldsymbol{x}^{t:t+L}$. In such cases, it is proposed the addition of a new hidden state to the current As-HMM. Suppose that the current model has parameter $\boldsymbol{\lambda}^0 = \{\boldsymbol{A}^0, \boldsymbol{B}^0, \boldsymbol{\pi}^0\}$ with $N \in \mathbb{N}$ hidden states. A new prior model is generated with parameter $\boldsymbol{\lambda}^1 = \{\boldsymbol{A}^1, \boldsymbol{B}^1, \boldsymbol{\pi}^1\}$ with $N + 1$ hidden states. The parameter $\boldsymbol{\pi}^1$ is built as:

$$\boldsymbol{\pi}^1 = [\boldsymbol{\pi}^0 | 0] \tag{7.1}$$

or append a 0 at the right of the vector $\boldsymbol{\pi}^0$. For the prior of $\boldsymbol{A}^1$, the following matrix

$\boldsymbol{C} = [c_{ij}]_{i,j=1}^{N+1}$ is built:

$$\boldsymbol{C} = \left[\begin{array}{ccc:c} & & & 0 \\ & & & 0 \\ & \boldsymbol{A}^0 & & \vdots \\ & & & 0 \\ & & & c_{N,N+1} \\ \hdashline \frac{1}{N+1} & \frac{1}{N+1} & \cdots & \frac{1}{N+1} \end{array}\right] \tag{7.2}$$

The value $c_{N,N+1} = 0.01$ is chosen for the EM to be aware that there is a non-null probability of transition among the current known hidden states to the latest discovered hidden state. The prior transition matrix $\boldsymbol{A}^1 = [a_{ij}^1]_{i,j=1}^{N+1}$ for the new model is $a_{ij}^1 = c_{ij} / \sum_{j=1}^{N+1} c_{ij}$ and $\boldsymbol{B}^1 = [\boldsymbol{B}^0 | b_{N+1}(\boldsymbol{x}^t)]$. It is assumed a priori that $b_{N+1}(\boldsymbol{x}^t)$ is represented by a naïve-Bayes graph with $p_{(N+1)m} = 0$ for $m = 1, ..., M$. The prior parameters of the corresponding new linear Gaussian Bayesian network are set as follows:

$$\begin{aligned} \beta_{(N+1)m0} &= \frac{1}{L} \sum_{j=t}^{t+L} x_m^j \\ \sigma^2_{(N+1)m} &= |\max_{j=t,...,t+L} \{x_m^j\} - \min_{j=t,...,t+L} \{x_m^j\}| \end{aligned} \tag{7.3}$$

During the learning optimization phase of $\boldsymbol{\lambda}^1$, the optimization exposed in Section 5.2 is restricted to the parameters $\{\boldsymbol{\beta}_{(n+1)m}\}_{m=1}^M$, $\{a_{(N+1)j}\}_{j=1}^{N+1}$ and $\{a_{j(N+1)}\}_{j=1}^{N+1}$. In this manner, the information obtained in previous hidden states is saved and the model is updated to explain the novel incoming data.

Although a renovation of the model is performed with the new data, the newly learned parameter $\boldsymbol{\lambda}'$ must be better than the current parameter $\boldsymbol{\lambda}^0$. Therefore, a condition in the BIC score for a new model to be valid is imposed, i.e., if $\boldsymbol{x}^{t:t+L}$ is the current data window, then the new model or parameter is accepted if $\text{BIC}(\boldsymbol{x}^{t:t+L}|\boldsymbol{\lambda}') < \text{BIC}(\boldsymbol{x}^{t:t+L}|\boldsymbol{\lambda})$.

Finally, whenever a new hidden state is added, the base health indexes need to be updated, but that will be explained in Section 7.1.3.1.

7.1.2 Novelty detection

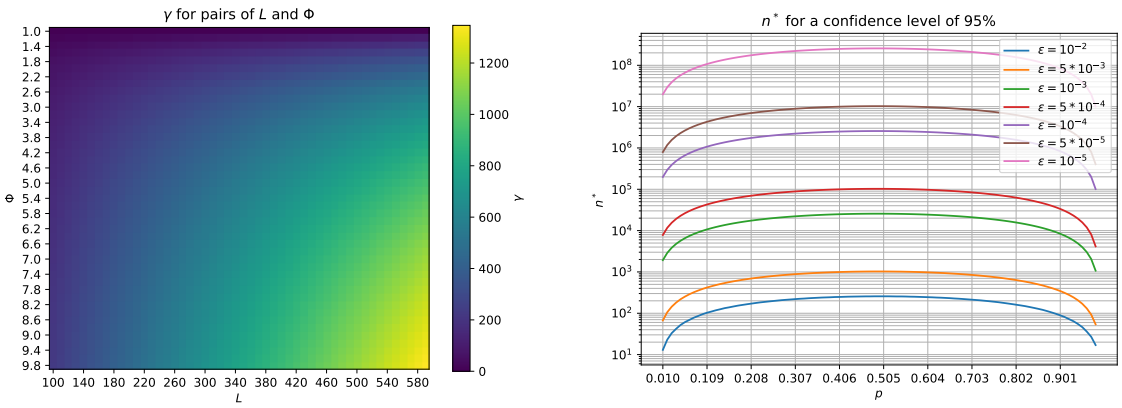
In Table. 7.1 the list of hyper-parameters used by this algorithm are mentioned. In the following subsections, they are detailed and it is explained how to tune and interpret them. In Appendix C a sensibility analysis is done over such hyper-parameters to observe their effect on the health index and RUL.

7.1.2.1 Anomaly detection

Set $L \in \mathbb{N}$ a window size which defines chunks of data $\boldsymbol{x}^{t:t+L}$. Assume that such window is sliced after $\triangle L \in \mathbb{N}$ data arrive. For the anomaly detection procedure, the Page sequential detection scheme is used (see Section 2.4.1). Suppose that an As-HMM with parameter $\boldsymbol{\lambda}$

Parameter	Set Value	Description
L	128	Length of the processing window
ΔL	10	Window slice
Φ	3	Maximum ratio of BIC difference
γ_1	$128\ln(3)$	Threshold in Page test to detect outliers
ϵ	1×10^{-2}	Maximum error of population percentage
p	0.1	Maximum enabled population of outliers
γ_2	0.05	Reliability of the population estimation
ζ	-2.5	Threshold for RUL prediction

Table 7.1: List of hyper-parameters used by the algorithm



(a) (b)

Figure 7.2: (a) Sensitivity of γ_1, for different values of L and Φ. (b) Sensitivity of n^* for different values of p and ϵ for a fixed level of confidence of γ_2

has been learned at time t. For this application, the Page-test funtion s is defined as follows:

$$s_t^l(\boldsymbol{x}^{t:t+l\triangle L+L}) := \frac{1}{l}\sum_{j=0}^{l} \mathrm{BIC}(\boldsymbol{x}^{t+j\triangle L:t+j\triangle L+L}|\boldsymbol{\lambda}), \tag{7.4}$$

In this scenario, l is the number of times the window has been sliced since the last model update. Since the BIC score is decreasing monotonous with the penalized fitness, increasing anomalous patterns are undesirable and must be recognized. Therefore, the increasing scheme detection of Page is used, see Eq. (2.65). Given a decision parameter $\gamma_1 > 0$, which indicates the maximum permissible deterioration in BIC, the test for this case is formulated as follows:

$$r_l = \begin{cases} 1 & \text{if } s_t^l(\boldsymbol{x}^{t:t+l\triangle L+L}) - \min_{j=0,\ldots,l-1} s_t^j(\boldsymbol{x}^{t:t+j\triangle L+L}) > \gamma_1 \\ 0 & \text{if } s_t^l(\boldsymbol{x}^{t:t+l\triangle L+L}) - \min_{j=0,\ldots,l-1} s_t^j(\boldsymbol{x}^{t:t+j\triangle L+L}) \leq \gamma_1 \end{cases} \tag{7.5}$$

Notice that the log-likelihood of the model could be used instead of the BIC for the Page test. Nevertheless, since the BIC is used during the training phase to determine the best set of context-specific Bayesian networks, this score is also used for anomaly detection for the sake

of consistency. The hyper-parameter γ_1 is set as $\gamma_1 = L\ln(\Phi)$ where Φ can be seen as the maximum permissible quotient of likelihood per testing observation over the likelihood per training observation. Fig. 7.2 (a) shows the value of γ_1 for different combinations of L (axis X) and Φ (axis Y). The greater the value of γ_1, the less the algorithm will declare deviated data as abnormal; whereas a lower value of γ_1 implies that little deviations in data will be considered as abnormal. Finding an equilibrium in this threshold is crucial for concept-drift detection. In Fig. 7.2 (a), it can be observed that after $\Phi > 1.8$, the change in L has more effect on the value of γ_1, but not extreme values are obtained. Therefore, in this study, $\Phi = 3$ is chosen as an intermediate point from the previous discussion. If $r_l = 1$, the chunk of data $\boldsymbol{x}^{t+l\triangle L:t+l\triangle L+L}$ is considered an anomaly for the model. In this chapter, $\triangle L = 10$ and $L = 128$ are fixed from previous experiments. The fitness and training speed of the models were maximized when these parameters were acknowledged.

7.1.2.2 Novel concept drift detection criterion

The detection of a single anomaly is not enough to determine a novel trend in the data. Nevertheless, if many outliers are seen in a time window, the possibility of observing a novel trend is likely. Assume that the probability of observing anomalies in the last n^* BICs is p, and $\hat{p}_{n^*}$ is the estimation of such probability. To determine the length n^*, the sampling problem is used, i.e., given an error $\epsilon \in (0,1)$, a proportion $p \in (0,1)$ and a precision $\gamma_2 \in (0,1)$, n^* must be found such that:

$$P(|\hat{p}_{n^*} - p| < \epsilon) > 1 - \gamma_2. \tag{7.6}$$

Assuming that for the current model the normal level of anomalies is $\hat{p}_{n^*} \leq p$, using the Chernoff bounds as in Section 2.4.2, n^* must be at least:

$$n^* > \frac{-\ln(1-\gamma_2)}{D_{KL}(p-\epsilon||p)}. \tag{7.7}$$

In Fig. 7.2 (b) the sensitivity of the bound n^* given by the Chernoff bounds for a precision level of $\gamma_2 = 0.05$ for different levels of ϵ and p is shown. As can be seen, the shorter the value of ϵ, the larger is n^* to ensure the bound. It is also noticeable that the curves are symmetric i.e., for a fixed ϵ and γ_2, the value n^* is the same for p and $1-p$. Since the goal is to detect novel trends as soon as possible, it is preferable to choose p closer to zero instead of one, and also greater values for ϵ. Therefore, in this study $p = 0.1$, $\epsilon = 10^{-2}$ and $\gamma_2 = 0.05$, in this manner a window size of $n^* = 87$ is obtained.

If R outliers are observed from the Page sequential test in a window of size n^*, $\hat{p}_{n^*} = R/n^*$ is computed. If $\hat{p}_{n^*} > p$, then the initial assumption is violated and a novel pattern is detected. Hence, an updating process of the HMM is performed.

7.1.3 RUL prediction

7.1.3.1 Proposed health index

It is plausible to think that the earlier the ball bearing is measured and the lower the vibration amplitudes are, the healthier the ball bearing is. Therefore, the first learned HMM from $\boldsymbol{x}^{0:L}$ is taken as a healthy model, and its parameters are used as a base health index. The first As-HMM is trained with only one hidden state, the emission probability $b_1(\boldsymbol{x}^t)$ can be transformed from a linear Gaussian Bayesian network into a MVN distribution (see, Section 2.1.3) with mean vector $\boldsymbol{\mu}_1 = (\mu_{11}, ..., \mu_{1M})$ and covariance matrix $\boldsymbol{\Sigma}_1 = \{\Sigma_{1jk}\}_{j,k=1}^{M}$. Define $\mu_{hm} := \mu_{1m}$ and $\Sigma_{hm} := \Sigma_{1mm}$ as the base health parameters of the ball bearing, compute q^t or the estimated hidden state for the observation $\boldsymbol{x}^t$ with the Viterbi algorithm; the following set of health indexes are defined:

$$HI_m^{\dagger}(\boldsymbol{x}^t) = \left\{ -\log\left(\frac{\mu_{q^t m}}{\mu_{hm}}\right), -\log\left(\frac{\Sigma_{q^t mm}}{\Sigma_{hm}}\right) \right\} \quad m = 1, ..., M. \tag{7.8}$$

Whenever a new hidden state is added (suppose it is the $N+1$ hidden state), the following base health assignment is performed:

$$\begin{aligned} \mu_{(N+1)m} &\rightarrow \mu_{hm} \text{ if } \mu_{hm} > \mu_{(N+1)m}, \quad m = 1, ..., M \\ \Sigma_{(N+1)mm} &\rightarrow \Sigma_{hm} \text{ if } \Sigma_{hm} > \Sigma_{(N+1)mm}, \quad m = 1, ..., M \end{aligned} \tag{7.9}$$

The idea behind this health index assignment is that the higher the mean and variance of fundamental-frequencies-amplitudes, the more evident the ball bearing wear. Finally, a global health index is defined as:

$$HI^{\dagger}(\boldsymbol{x}^t) = \min \bigcup_{m=1}^{M} HI_m^{\dagger}(\boldsymbol{x}^t). \tag{7.10}$$

This global health index takes the worst health index obtained among all the features. Some final considerations: the latest q^t detected is saved in order to be used in further algorithms such as Viterbi algorithm or forward-backward algorithm. The reason behind this is that during the traditional forward-backward algorithm or Viterbi algorithm, the parameter $\boldsymbol{\pi}$, which indicates the participation of the initial distribution, is always used. Nevertheless, since it is possible that further incoming chunks of data $\boldsymbol{x}^{t:t+L}$ do not contain evidence of the first learned hidden state, the parameters $\{a_{q^t i}\}_{i=1}^{N}$ must be used as initial distribution for the algorithms. Finally, in order to obtain smoother health index curves, the health index historic $\{HI^{\dagger}(\boldsymbol{x}^l)\}_{l=1}^{t}$ is passed through a moving average filter of order twenty.

7.1.3.2 RUL estimation

In this thesis, a time-dependent model is created to explain and predict the global health index. A regression model is used after the first concept drift is detected:

$$HI^{\dagger}(t) := a_0 + a_1 t + a_2 t^2 + w(t), \tag{7.11}$$

where $w(t)$ is a zero mean normal error term. The parameters of the regression models are estimated as in an ordinary least squares problem.

Once the regression has been estimated, RUL^t is determined as follows: find t_f such that $HI^{\dagger}(t_f) = \zeta$, where $0 > \zeta > -\infty$ is a maximum permissible magnitude order deviation from the good health estimation. In this study, $\zeta = -2.5$ or, in other words, a failure happens for this health index when the global process trend is 2.5 orders of magnitude away from the healthy trend. Nevertheless, depending on the mechanical application, the threshold may be changed. In applications where little deviations from the normal states imply a malfunction, the threshold must be closer to zero; otherwise, further away from zero. The RUL at time t is computed as:

$$\mathrm{RUL}^t = t_f - t. \tag{7.12}$$

If $HI^{\dagger}(t_f) \neq \zeta$, $\forall t$, then $\mathrm{RUL}^t = \infty$ and this can be interpreted as a non-degradation in the health of ball bearings.

7.1.4 Theoretical computational complexity

In Table 7.2 theoretical bounds using big O notation are given for an iteration of the proposed algorithm. These bounds assume that the AR and context-specific Bayesian networks are dense. For the sake of space $K = p^* + M$. For the *Novelty detection* box, the complexity lies in the BIC computation, whereas the Page sequential test and Chernoff bounds have $O(1)$ complexity. The *Model update* box complexity relies principally on the SEM algorithm. Finally, for the *RUL regression* box, the complexity comes from the Viterbi algorithm and the $HI^{\dagger}$ computation.

The bound for the *Model update* box suggests that the algorithm is too complex to perform in online environments. Therefore, this box is further analyzed in Section 7.3.3.1 to analyze its behavior with real data.

Phase	Complexity
Novelty detection	$O(LN(NK + N^2))$
Model update	$O(M^2K^2(K(L + K) + NL) + LN(N^2 + MK))$
RUL prediction	$O(T + N(MK + M^2 + L))$

Table 7.2: Computational complexity of one iteration of the proposed online algorithm

7.2 Experimental setup

To evaluate and see the capabilities of the proposed methodology, two data-sets were used. The first one comes from the online FEMTO repository which has ball bearing RTF data Nectoux et al. [2012]. The second dataset comes from a mechanical setup by Aigura IIoT, where ball bearings are run to failure. In this manner, the capabilities of the proposed method for common data (FEMTO) and specific daily data (Aingura IIoT) are proven.

The models WPD-HMMOcak et al. [2007], AHMM Yu [2017] and APCMD Wu et al. [2019], are used for comparison purposes in the case of the FEMTO dataset. AHMM is used for comparison since it can be considered as a simple application of HMM for online analysis where no novelty detection or model update is considered; WPD-HMM is an HMM where the model is updated whenever a new instance arrives. However, in none of these, there is a description or algorithm to compute the RUL. In particular, no HMM model with no RTF data assumption for RUL prediction was found. For that reason, APCMD is used to compare RUL prediction.

The previous methodologies use a different definition of HI, which hardens the comparison task. In the case of WPD-HMM Ocak et al. [2007], the HI is the log-likelihood/BIC of incoming data being evaluated by an HMM learned with normal or healthy data. The closer the log-likelihood/BIC to zero, the better the model; if the log-likelihood/BIC decreases/increases, the model fitness is worse and abnormal data is being processed. AHMM Yu [2017] proposed a HI given by the Cauchy-Schwarz correlation of the AHMM dynamic evolution. Here a correlation of 100% implies no change in the current distribution or model and 0% deduces a total drift from the current distribution or model. APCMD Wu et al. [2019] proposed a HI based on the Mahalanobis distance of a reduced set of features after being processed by a principal component analysis (PCA). In this case, a distance of 0 implies no change in distribution, whereas a big distance suggests a drift from the normal data.

The proposed methodology will be compared as follows: In the case of WPD-HMM, the BIC curve works as HI; then it is compared with the one obtained by the proposed methodology. For AHMM, as their methodology uses a Cauchy-Schwarz correlation as health index, it is compared with $HI^{\dagger}$. APCMD and the proposed methodology are the only ones that can generate RUL predictions; therefore their results in HI and RUL are contrasted.

7.2.1 FEMTO dataset

The benchmark used to validate the model comes from the "FEMTO Bearing Data Set" Nectoux et al. [2012][1]. This dataset consists of RTF bearings data under critical rotational speed and load conditions. With these conditions, the bearings are forced to fail faster. The idea is to estimate the ball bearing health state up to failure and its RUL. Table 7.3 shows a description of the dataset. In this dataset, a ball bearing fails when the accelerometer records are above 20 g, where g is the gravity acceleration at sea level ($9.7805\frac{m}{s^2}$).

[1]https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/ online status on December 12, 2022

Condition	Label	Purpose	# Samples
1	Bearing1_1	Training	2803
1	Bearing1_2	Training	871
1	Bearing1_3	Testing	1802
1	Bearing1_4	Testing	1139
1	Bearing1_5	Testing	2302
1	Bearing1_6	Testing	2302
1	Bearing1_7	Testing	1502
2	Bearing2_1	Training	911
2	Bearing2_2	Training	797
2	Bearing2_3	Testing	1202
2	Bearing2_4	Testing	612
2	Bearing2_5	Testing	2002
2	Bearing2_6	Testing	572
2	Bearing2_7	Testing	172
3	Bearing3_1	Training	515
3	Bearing3_2	Training	1637
3	Bearing3_3	Testing	352

Table 7.3: Training and test data sets of the FEMTO dataset

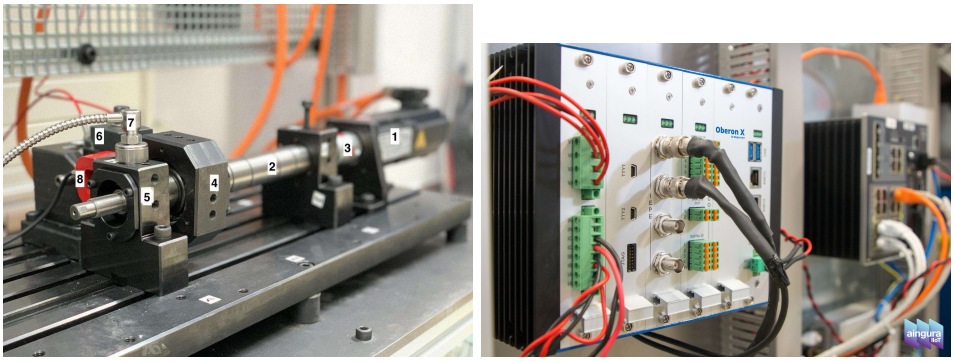
Since our model assumes that a data stream is being received and no previous RTF data is available, the training-testing specification is ignored and those datasets which show evidence of degradation and not a sudden failure are used. Bearings with labels Bearing1_1, Bearing1_3, Bearing2_1 and Bearing2_2 show degradation among the previous mentioned bearings. For this thesis, the bearings Bearing1_1, Bearing1_3 and Bearing2_2 are used to test the proposed methodology.

7.2.2 Mechanical setup dataset

The purpose of this ball bearing testing set-up is to monitor vibrations overtime during ball bearing useful life. The experimental set-up is shown in Figure 7.3a. This testbed has a Bosch IndraDyn MS2N synchronous servomotor $\boxed{1}$ that guarantees required speed during testing, a shaft $\boxed{2}$ with different ball bearing clamping positions and an elastic coupling $\boxed{3}$ to the servomotor.

The testbed has three ball bearing supports, one for the axial force actuator $\boxed{4}$ and two for support. The system is designed to affect the outside ball bearing $\boxed{5}$ useful life applying an axial force. The force actuator is a screw-based system fitted with a load sensor that measures the applied force $\boxed{6}$. An IMI 607A61 accelerometer with a sensibility of 10.2 mV and vibrational-signal detection range between 0.5-10 kHz. $\boxed{7}$ is used to measure vibrations, and a thermocouple $\boxed{8}$ to monitor temperature and guarantee testbed integrity.

Figure 7.3b shows the data acquisition and pre-processing edge computing device called Computing Module Aingura Insights (CMAI). The CMAI compute node is an embedded system powered by a ZU3 Zynq® Ultrascale+™ System on Chip (SoC) housing four cores



(a) (b)

Figure 7.3: Experimental testbed: (a) Ball bearing RTF testbed details. (b) CMAI

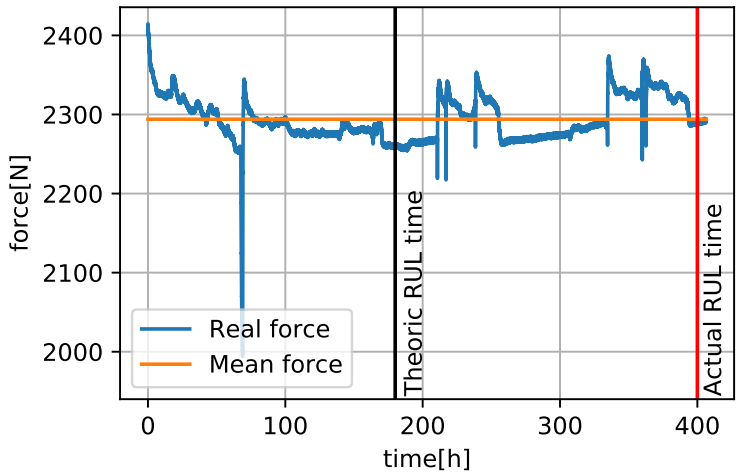


Figure 7.4: Observed force evolution given by the actuator

Cortex-A53 and a programmable logic. The node is customized with add-on modules collecting data from different sensors, and it can be coupled with other nodes to improve its computational capabilities.

For our experiments, four channels are implemented for accelerometer readings up to 19.5 kHz and sensor fusion capabilities to guarantee signal synchronization. Data collected are stored in files as comma-separated values (CSV).

The CMAI is used to acquire, condition and pre-process the acceleration signal at a sampling rate of 19.5 kHz with 0.5 ns of maximum jitter. In a production setup, the CMAI has enabled processing capabilities to work online with the proposed technique. For validation and complete control over the proposed pipeline, sensor data is stored into CSV files for offline analysis, simulating the data-stream as in an online environment. The implementation of the model has been written in C++, compiled with GCC 10.2, and executed using a single core of the ZU3 SoC. Ball bearing data are collected and pre-processed by a single CMAI.

For our test, a radial force of 2.3-2.4 kN and a rotational speed of 3180 RPM (53 Hz)

were selected from the traditional mechanical analysis theory Budynas et al. [2011] in order to run an Eco 6004-2RS ball bearing to failure between 180 hours (assuming 2.3 kN) and 156 hours (assuming 2.4 kN). The force measured profile is shown in Fig. 7.4. The mean force was around 2.3 kN which implies that the theoretical RUL was close to 180h. However, the total testbed operation time was over 400 hours, meaning 2.25 times beyond the theoretical useful life. As a safety measure, since the ball bearings operation time overpassed by several hours the theoretical RUL predictions given by the provider, the acquisition was stopped when audible signals of degradation were evident. Finally, a signal smoothing processing was applied: a root-mean-square was computed every 50 samples from the FE box in Fig 7.1, equivalent to collapsing 85 seconds of data to a single data instance.

The CMAI was configured to keep the amplitudes of fundamental frequencies and discard the rest of vibrational data. Health index ($HI^{\dagger}$) and RUL predictions were stored.

7.3 Results

7.3.1 Time analysis

In this section a time consumption of the proposed algorithm is shown. Regarding the feature extraction (FE) box of Fig. 7.1, the CMAI required at least 1.6 seconds of measurements for a reasonable granularity in the signal spectrum in order to obtain the desired (BPFO, BPFI, BSF, FTF) frequencies with precision. The signal processing and feature extraction follows the methodology proposed by Bechhoefer [2005] (See 2.5). After the data acquisition, the CMAI required 1.5 seconds to extract the desired features.

In Table. 7.4, the mean and standard deviation of the time execution algorithm of the three boxes in Fig. 7.1, say: *novelty detection*, *model update* and *RUL prediction* are shown for each tested data-set. Note that for the FEMTO data-sets, the maximum mean time needed for the *Novelty detection* process it was 1.16ms, for the *Model update* was 6.03ms and for *RUL prediction* it was 2.44ms. On the other hand, for the mechanical set-up, it is observed that the mean times for the *novelty detection* were 8.73 times higher than in the case of the FEMTO datasets, for the *model update* 15.1 times higher and for *RUL prediction* it was 9.37 times higher. This can be explained due to the length of the data and the higher number of hidden states. Finally, it is observed that the standard deviations of the times for all the cases were lower than one second which gives evidence of relatively stable algorithms.

These time results show that the algorithm can achieve a fair time response for an online data stream environment. Nonetheless a more detailed analysis is given in Section 7.3.3.1

7.3.2 FEMTO results

7.3.2.1 Bearing1_1 results

Fig. 7.5 summarizes the results obtained for the data-set Bearing1_1. The BIC score from the WPD-HMM methodology is drawn in Fig. 7.5a which is used as a health index. It can be observed that WPD-HMM obtains results different from those obtained by the online

Dataset	Phase	$\bar{t}$(ms)	S_t(ms)
Bearing_11	Novelty detection	0.960	0.713
	Model update	4.291	32.474
	RUL prediction	2.188	1.429
Bearing_13	Novelty detection	1.160	1.005
	Model update	6.025	39.298
	RUL prediction	2.440	2.201
Bearing_22	Novelty detection	0.982	0.417
	Model update	3.933	23.100
	RUL prediction	1.610	0.746
Set-up	Novelty detection	10.130	7.015
	Model update	91.049	292.751
	RUL prediction	22.884	16.450

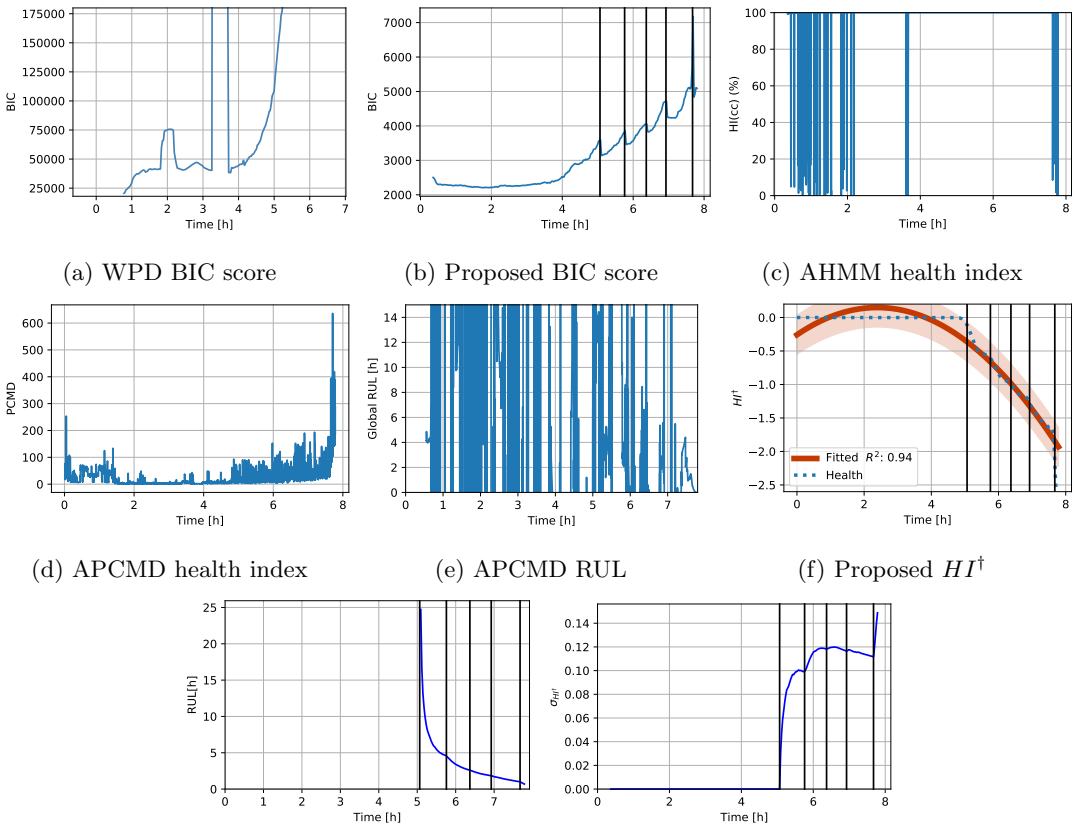
Table 7.4: Execution times statistics of the different phases of the algorithm in the CMAI, for the different datasets

As-HMM which are displayed in Fig. 7.5b. In particular, it can be noticed that, arriving at the fourth operational hour, the BIC score in WPD-HMM had a discontinuity; later it returned back to a lower BIC score, but followed by exponential growth. For the WPD-HMM methodology, it can be argued that, before the fourth hour, an accelerated degradation process to failure can be observed. On the other hand, the proposed methodology uses a novel detection technique. When a novel trend was detected at time t, a vertical black line was displayed in the plot. It can be said that whenever a novel trend was detected, the BIC score was improved and drastic changes in BIC (as observed in the case of WPD-HMM) were avoided. However, an essential controlled growth in BIC could be observed after the fourth hour.

The health index in the case of AHMM, which can be seen in Fig. 7.5c, reveals a noisy behavior at the beginning and at the end of the ball bearing life which is not informative at all. In particular, the early decays to 0 would imply an early failure, which is not true. $HI^{\dagger}$ drawn on Fig. 7.5f remained unchanged until the first novel trend was detected. After that, $HI^{\dagger}$ decreased and showed evidence of degradation.

In this experimental set-up, the uncertainty of the RUL was exposed in Fig. 7.5h with the standard deviation of the health index prediction. This uncertainty is the level of fidelity of the health index regression; the higher this measure, the worse the fitness of the regression for RUL prediction. Regarding Bearing1_1, the uncertainty appeared after the first concept drift as expected. Note that the uncertainty or standard deviation was around 0.12 orders of magnitude, far from half an order of magnitude, which indicated a fair regression for RUL prediction. This can be visualized in the health index regression shaded curves in Fig. 7.5f It is remarkable to say that the uncertainty varied with time, because the regression was updated whenever a new $HI^{\dagger}$ instance was computed.

The proposed methodology and APCMD are capable of prognosis. Both use a regression process to predict the health index. However, in the case of APCMD, the health index is a



(a) WPD BIC score (b) Proposed BIC score (c) AHMM health index

(d) APCMD health index (e) APCMD RUL (f) Proposed $HI^\dagger$

(g) Proposed RUL (h) Regression uncertainty

Figure 7.5: Results of supervising Bearing1_1 with different methodologies: (a) shows the WPD BIC score evolution, (b) draws the As-HMM online model BIC score evolution, (c) gives the AHMM health index, (d) pictures the APCMD health index, (e) displays APCMD RUL prediction, (f) shows $HI^\dagger$ results, (g) is the obtained RUL curve from the proposed methodology, (h) is the standard deviation of the proposed health index regression

Mahalanobis distance based on the PCA and, in the case of the proposed methodology, $HI^\dagger$ is based on logarithms of ratios. Both methodologies require a threshold on the health index to determine the RUL. The threshold in the case of APCMD was set to 200; this value was set after observing the evolution of the Mahalanobis distance exposed in Fig. 7.5d. In the case of APCMD, a noisy RUL prediction during almost all the ball bearings life is drawn in Fig. 7.5e. In particular, it can be observed that at early times the RUL was zero, which would imply an early failure, which again was not true. Nevertheless, at the end of the process, after the seventh hour the accuracy was improved, for the given threshold. In the case of the proposed methodology, the RUL, pictured in Fig. 7.5g, was only computed after the first concept drift; after that, the RUL went towards zero. It is worth noting that the first predictions overestimated the RUL, and later, the regression gave better predictions.

From the point of view of applicability, a maintenance engineer will use the actionable insight from the proposed methodology as a decision support element to plan ball bearing maintenance. It is important that the actionable insight provided is relevant where the ball

bearing is critical to operation, with a high risk of stopping the machine or production line if an unexpected failure occurs. Therefore, if the engineer on charge observes a decrease in $HI^{\dagger}$, the engineer should begin to plan a controlled maintenance stop with haste as indicated by the indicator trend.

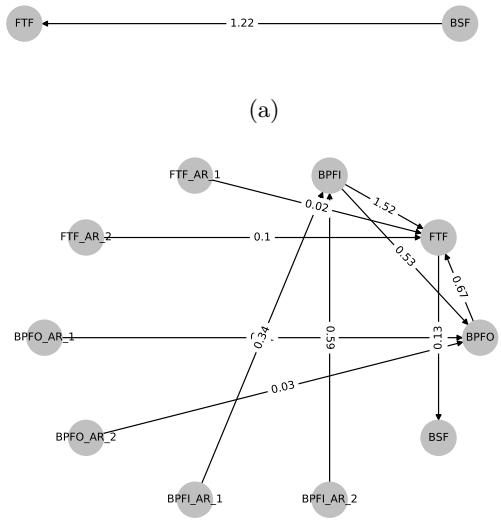
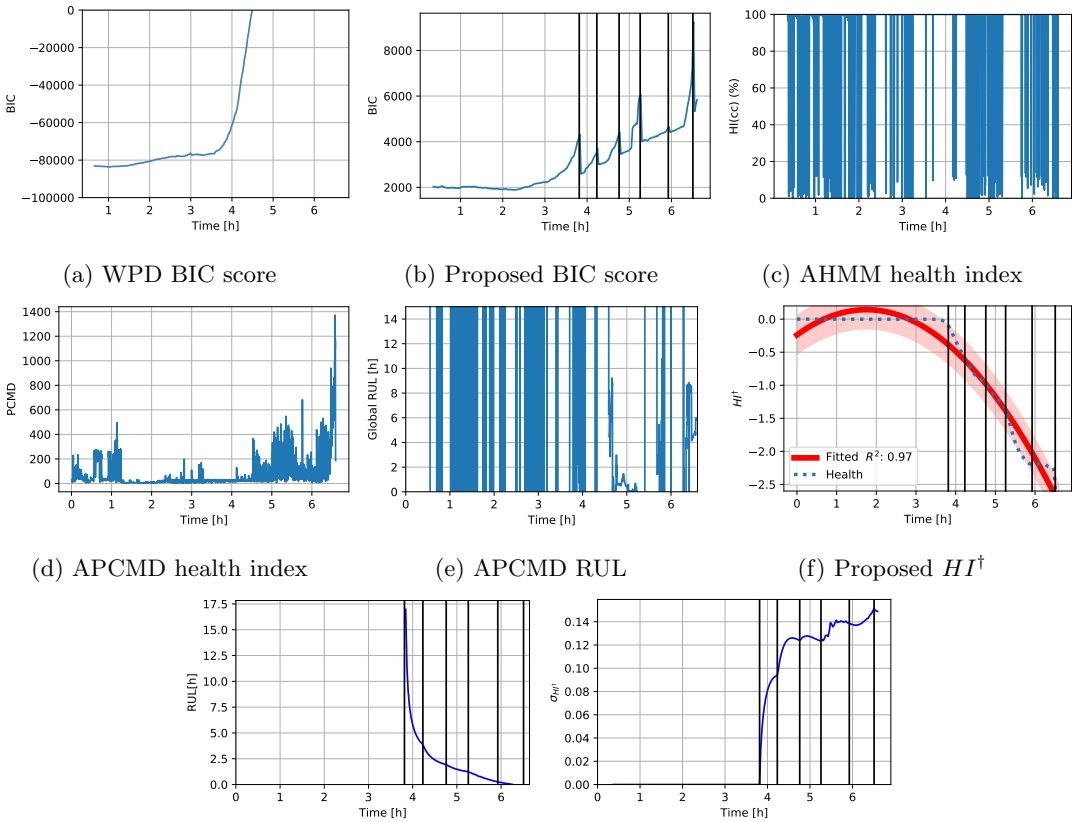


Figure 7.6: Learned Bayesian networks from Bearing1_1

As stated before, the proposed methodology is capable of giving an explanatory model, where temporal and instantaneous probabilistic relationships may appear. Fig. 7.6 displays a pair of generated Bayesian networks from the learned hidden states. Fig. 7.6a shows the Bayesian network generated from a low degradation state, whereas Fig. 7.6b draws a Bayesian network from a degenerated state. Observe that, in a low level of degradation, in this case, few probabilistic relationships appear, whereas in a degraded state, in this case, more relationships appear. Each arc represents a probabilistic dependency; in the case of a degraded state, it can be observed that the BPFI frequency-amplitude affects the BPFO and the FTF frequency-amplitudes; or in other words, in this ball bearing, in a degraded state the ball bearing inner ring behavior drives the ball bearing outer ring and cage. Following the structure of the Bayesian network, it can also be observed that some AR dependencies appear e.g., the BPFI frequency-amplitude is affected by two of its previous values.

7.3.2.2 Bearing1_3 results

Fig. 7.7 shows the results obtained for the dataset Bearing1_3. The BIC score from the WPD-HMM methodology is pictured in Fig. 7.7a. In this case is is observed that after time 4 h, the BIC score grew abruptly, which indicates an accelerated degradation and failure. If it is compared with the BIC score obtained from the proposed methodology Fig. 7.7b, a concept drift appeared at the fourth operational time and the BIC grew from this point in a



(a) WPD BIC score (b) Proposed BIC score (c) AHMM health index

(d) APCMD health index (e) APCMD RUL (f) Proposed $HI^\dagger$

(g) Proposed RUL (h) Regression uncertainty

Figure 7.7: Results of supervising Bearing1_3 with different methodologies: (a) shows the WPD BIC score evolution, (b) draws the As-HMM online model BIC score evolution, (c) gives the AHMM health index, (d) pictures the APCMD health index, (e) displays APCMD RUL prediction, (f) shows $HI^\dagger$ results, (g) is the obtained RUL curve from the proposed methodology, (h) is the standard deviation of the proposed health index regression

controlled manner with the appearance of more concept drifts. This difference evident that our proposed methodology is capable of describing, in a refined manner, the ball bearing degradation and failure states.

In the case of the AHMM health index, which is pictured in Fig. 7.7c, this time it shows a noisy behavior during all the ball bearing operational time. Although the Cauchy-Schwarz distribution correlation is easy to interpret (0% implies zero correlation between distributions and 100% indicates perfect correlation), in this case, due to the noisy behavior and jumps between 100% and 0%, no insights can be extracted from the health index. Meanwhile $HI^\dagger$ in Fig. 7.7f, remained unchanged until the first novel trend was detected as in the case of Bearing1_1. After that, $HI^\dagger$ decreased and showed evidence of degradation until it reached the failure threshold.

Now, concerning RUL, APCMD as before obtained a noisy prediction, as shown in Fig. 7.7e, of the ball bearings RUL. It indicates failure during the first hours of the bearing operation, which is not true. Even during the last hours of the ball bearing operation, the

RUL prediction was highly inaccurate. On the other hand, the proposed model RUL prediction (see Fig. 7.7g) showed predictions only after the first concept drift was detected. As in the case of Bearing1_1, at the first prediction times, the RUL was overestimated; however, as time went, the RUL prediction became more accurate until $HI^{\dagger}$ passed the failure threshold. Since the proposed methodology exposed a similar prediction as in the case of Bearing1_1, the maintenance policies that can be generated for the Bearing1_1 are also valid for this ball bearing (Bearing1_3).

In this case, it is observed that the uncertainty of the regression pictured in Fig. 7.7h used for RUL prediction was different from that exposed in Bearing1_1. Observe that the level of uncertainty did not pass over 0.15 orders of magnitude, which implies that the fidelity of the RUL was fair.

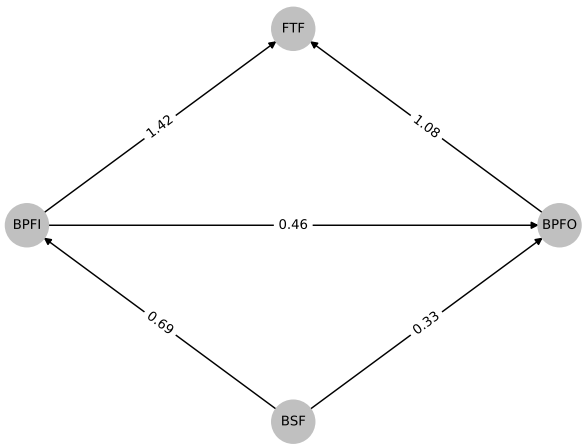
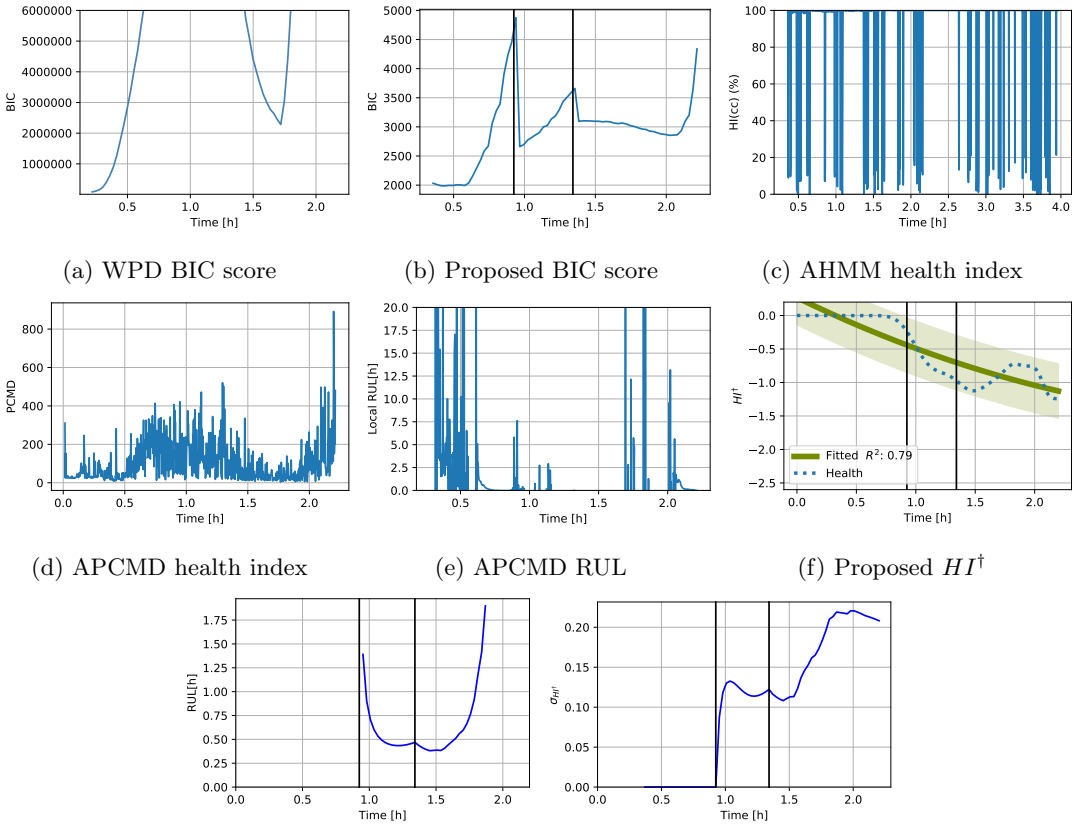


Figure 7.8: Learned Bayesian networks from Bearing1_3

For the sake of completeness, one of the learned Bayesian networks learned by the asymmetric hidden Markov model is shown. Fig. 7.8 displays one Bayesian network from an intermediate degraded state. For example, the FTF frequency-amplitude depends on the BPFI and BPFO frequency-amplitudes, the BPFI frequency-amplitude depends on the BSF frequency-amplitude, and the BPFO frequency-amplitude relies on the BPFI and BSF frequency-amplitude. From this graph, it can be said that the ball bearing balls are the ones leading the mechanical behavior of the ball bearing at an intermediate degradation level.

7.3.2.3 Bearing2_2 results

Fig. 7.9 shows the BIC score from the WPD-HMM methodology. It can be observed that, during the first hour of operation, the BIC score grew in an important manner, which for that methodology, implies an early fast degradation process in the ball bearing. On the other hand, the proposed methodology observed a concept drift during the first hour of operation; nevertheless, the evolution of the BIC score did not grow as uncontrolled as in the WPD-HMM methodology. It can be said, as in the case of Bearing1_1, that whenever a novelty



(a) WPD BIC score (b) Proposed BIC score (c) AHMM health index

(d) APCMD health index (e) APCMD RUL (f) Proposed $HI^\dagger$

(g) Proposed RUL (h) Regression uncertainty

Figure 7.9: Results of supervising Bearing2_2 with different methodologies: (a) shows the WPD BIC score evolution, (b) draws the As-HMM online model BIC score evolution, (c) gives the AHMM health index, (d) pictures the APCMD health index, (e) displays APCMD RUL prediction, (f) shows $HI^\dagger$ results, (g) is the proposed methodology RUL prediction, (h) is the standard deviation of the proposed health index regression

detection occurred, the BIC score was improved and this adaptation prevented the model from obtaining drastic changes in this score. However, in this case, a clear increasing trend in BIC was not seen as in the case of Bearing1_1 or Bearing1_3; on the contrary, from the first and a half hour to the second hour of operational time, a decrease in BIC was obtained.

The health index in the case of AHMM (Fig. 7.9c) reveals again a noisy behavior that does not provide any relevant information if it is compared to any of the other methodologies. In the case of APCMD, the health index (Fig. 7.9d) shows noisy growth indicating a worsening in the ball bearing condition. Later, the health index decreased between the first hour and a half to the second hour of operational time which implies improvement in the ball bearing condition. However, since this health index lies in the PCA projected space it is hard to tell if the improvement was significant or not. In the case of our methodology, the health index $HI^\dagger$ decreased after the first concept drift. Nonetheless, between the first hour and a half to the second hour of operational time (as the WPD and APCMD methodologies also detect) a slight (0.4 orders of magnitude) health improvement was observed; after that, $HI^\dagger$ decreased

again.

In terms of RUL curve, the APCMD obtained a noisy prediction during almost all the ball bearing life as shown in Fig. 7.9e. In this case, the local RUL had to be used instead of the global because it obtained poor results. Nevertheless, at the end of the process after the second hour of operation the accuracy was improved. Fig. 7.9g shows the obtained RUL curve from the proposed methodology. In this case, after the first novel detection flag, a decreasing tendency was observed. But, as mentioned before, an improvement in health was observed; this violates the second assumptions imposed in Section. 7.1. As a consequence of this violation, the prediction curve sign convexity changed, which could be translated into the increasing trend in the RUL curve. Although the RUL prediction was poor in this case, thanks to $HI^{\dagger}$, there was statistical evidence of bearing degradation.

The uncertainty of the RUL prediction in this case is shown in Fig. 7.9h. In spite that the RUL prediction in this case was worse when compared with the other datasets from FEMTO, the standard deviation of the prediction obtained fair values below a quarter of order of magnitude. However it is relevant to remark that the change in the sign of concavity in the health index after the first hour and a half affected drastically and negatively to the confidence of the prediction.

In this case, from the point of view of application, these actionable insights are also useful to explore failure related to wrong installation procedure for the ball bearing. That is, early problems could suggest that the ball bearing has problems with actual operating conditions and must be reinstalled as soon as possible to avoid unexpected failures.

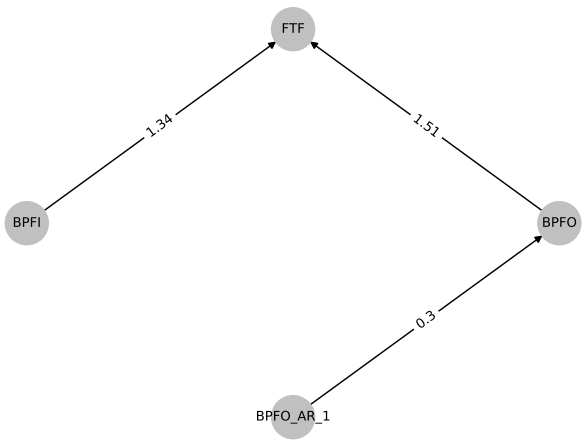


Figure 7.10: Learned Bayesian networks from Bearing 2_2

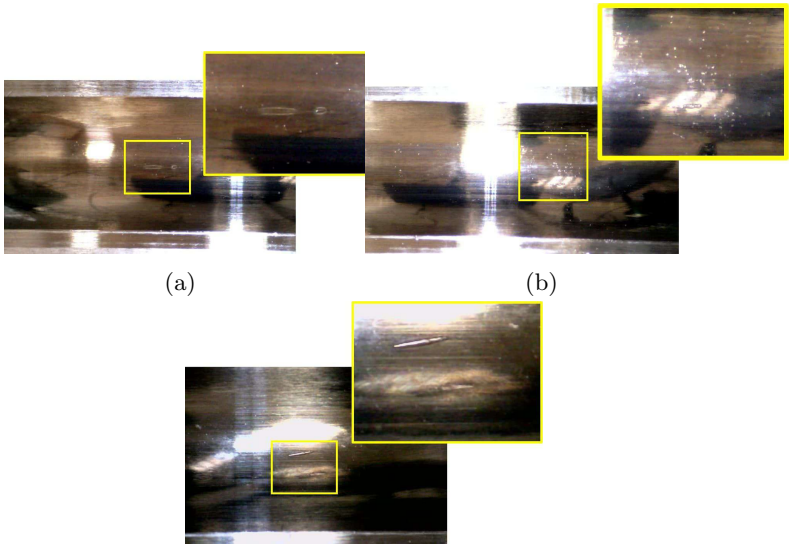
As in the case of Bearing1_1 and Bearing1_3, for each learned hidden state, context-specific Bayesian networks can be extracted. In Fig. 7.10 the resulting Bayesian networks from a degraded state is drawn. In this scenario, the FTF frequency-amplitude dynamics can be explained by the BPFI and BPFO frequency-amplitude. Also, the BPFO depends on one previous value. In this case, the inner and outer race lead the dynamic behavior at a

degraded state.

7.3.3 Mechanical setup results

In Fig. 7.11, the worn ball bearing inner ring is shown. Different degradation pieces of evidence are detected, such as in Fig. 7.11a, where a rolling surface indentation is found, usually started by small particles generated by the ball cage or the lateral seals. In Fig. 7.11b and 7.11c, porosity has developed over the rolling surface, related to the lubrication issues created by lateral seal failure and temperature. As shown in Fig. 7.12a, most of the time, the temperature was between 40 and 45°C, which is not enough to damage the ball bearing. However, this effect mixed with the failure of the seals can create lubrication problems in the balls.

Fig. 7.12b pictures the smoothed evolution of the ball bearings fundamental frequencies (BPFO, BPFI, BSF, FTF) extracted from the FE algorithm. From this picture, it can be observed that there is evidence of ball bearing degradation since the ball bearing frequency-amplitudes increase over time; in particular, the FTF frequency-amplitude exhibits a more significant positive trend. Nevertheless, by just looking, it is not possible to extract a clear health index or bearings RUL estimation. This enables the use of the proposed methodology.

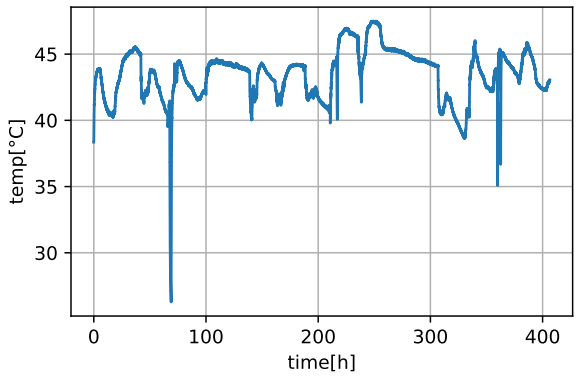


(a) (b)

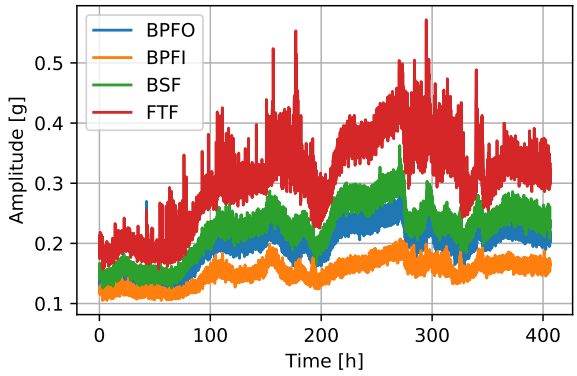
(c)

Figure 7.11: Test results: (a), (b), (c) show evidence of degradation in the inner ring

It is expected that the proposed methodology (Fig. 7.1) process data streams within embedded electronics. However, to analyze its performance, the batch SEM algorithm of the As-HMM Puerto-Santana et al. [2022b] is studied from the computational point of view inside the CMAI module. In this way, memory and time constraints and opportunities can be deduced for the model learning phase and can enable the model to be optimally employed in an online data stream.



(a)



(b)

Figure 7.12: a shows the Measured temperature evolution of the ball bearing. b shows the Evolution of the ball bearings fundamental frequencies

7.3.3.1 Execution time and performance analysis

The As-HMM implementation has been evaluated using the following configuration: 25 variables, a window size of 5000 samples, and 3 hidden states. The C++ code makes use of two external libraries, igraph v0.8.4 and openblas v0.3.13 and it has been compiled with GCC v10.2. All computational measurements presented in this section have been gathered on one CMAI.

The first step for evaluating the experimental setup has been to study the execution time of our optimized implementation of the As-HMM algorithm. Fig. 7.13 shows the phases of the algorithm: the color code represents the percentage of the execution time taken by each phase, from green, corresponding to short execution time, to red, corresponding to steps taking up to 35% of the total execution time. From Fig. 7.13, the reader can conclude that `Update` $\boldsymbol{\beta}$ `and` $\boldsymbol{\eta}$, `Forward-Backward`, `Update Temporal Mean`, `Update` σ are the phases taking more time, summing up 87% of the total execution time.

Fig. 7.14 shows the execution time of the four phases on a single Cortex-A53 of the CMAI.

Using hardware counters, it has been possible to count different types of instructions,

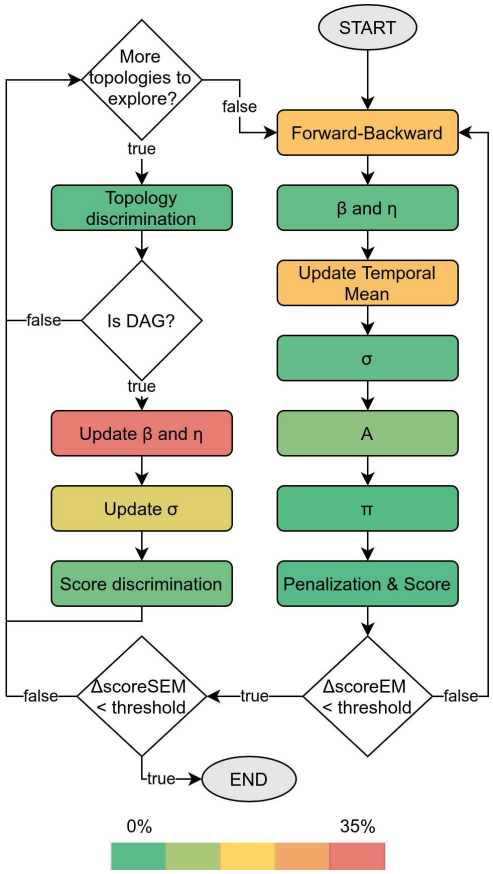


Figure 7.13: Steps of the As-HMM algorithm color coded depending on the relative execution time

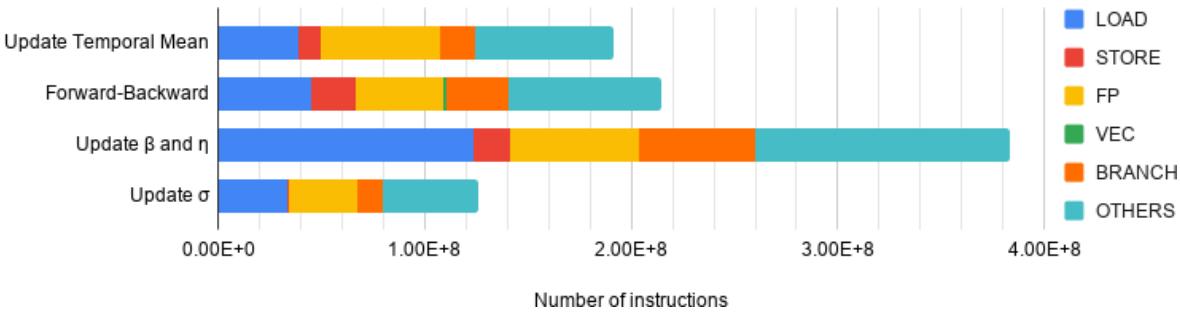


Figure 7.14: Instruction mix of the four most time consuming phases of the As-HMM algorithm

which are also reported in Fig. 7.14. The classes of instructions identified are: memory accesses (LOAD and STORE or LD and ST for short respectively); floating-point instructions (FP as counting the scalar instructions and VEC as counting the instructions executed by the single instruction multiple data (SIMD) unit of the core); branches and jumps (BRANCH); integer operations and other instructions for the book keeping of the program (OTHERS).

From a first analysis of the type of instructions, it appears that the compiler was not able to vectorize the code. There were, in fact, very few vector instructions (VEC) in all phases so that they can be ignored in the rest of our study. Also, the phase `Update` σ had almost

no store instructions: it had been verified that, indeed, the `Update` σ phase required loading on average more than 300 data to update/store one value to the memory.

	Instruction type per clock				
	IPC	LD + ST	FP	BRANCH	OTHERS
Update β and η	0.90	0.33	0.15	0.13	0.29
Forward-Backward	0.80	0.25	0.16	0.11	0.28
Update σ	0.74	0.20	0.20	0.07	0.27
Update Temporal Mean	0.69	0.18	0.21	0.06	0.24
Correlation with IPC	*1.00*	*0.99*	*-0.94*	*0.96*	*0.91*

Table 7.5: Density of the different type of instructions in four phases of the As-HMM algorithm

To understand if there are types of instructions that are harming the performance of our implementation, the density over time of each type of instruction has been computed. The global density of all instructions is called Instructions per Clock-Cycle (IPC), and it has been computed for each phase. The densities for all classes of instructions have also been computed: memory accesses (load and store instructions are aggregated, LD + ST) per cycle, floating-point (FP) per cycle, branch (BRANCH) per cycle, and other instructions (OTHERS) per cycle. All values are reported in Table 7.5. Phases are sorted by the decreasing value of the IPC, and the values of instruction densities have also been color-coded. Looking at the color gradients, it can be detected that there is a direct correlation between the IPC (yellow gradient) and all instruction densities (green gradients), except for the floating-point instructions. This means that increasing the number of floating-point instructions per unit of time implies a lower IPC resulting in performance degradation.

Table 7.5 highlights that there is a direct correlation between the IPC and the density of memory accesses (LD + ST): this result is counterintuitive. Since the memory is the slowest of the resources in a compute node, one would expect the opposite. For this reason, the number of accesses to memory that were misses in the two-level of caches of the Cortex-A53 core have been studied. The measured number of misses per kilo-instructions (MPKI) are between 1.56 and 7.63 in the L1 cache and between 0.01 and 0.09 in the L2 cache (that is the last level cache of the Cortex-A53). This means that almost all the memory accesses of our implementation are hit on cache, explaining the positive effect of memory instructions on the global IPC.

This effect depends on the configuration of the model: this would probably become worse, increasing the number of variables, hidden states, or the size of the window. Also, changing the configuration of the model would change the data structures storing the graph used for the model. The code is susceptible to changes in the data structures so that one can expect cache effects in the case of changes in the initial configurations.

The data structures of the model are sparse and have an arithmetic intensity between 0.06 and 0.15 Floating Point Operations per Byte (FLOPs/Byte). These low values of the

arithmetic intensity can be mitigated by reorganizing the data structures as adjacency lists. This has the benefit that the code can operate with more dense objects and maximize the cache reuse. As a price to pay, more integer instructions for pointer bookkeeping are needed. This is visible in Fig. 7.14 with the fact that almost 30% of the instructions of all phases are of type OTHERS.

7.3.3.2 Health states and RUL prediction

The weaknesses and strengths of the underlying model (As-HMM) of the proposed methodology have been explored and mentioned. The corresponding results from the methodology using a ball bearing are now shown.

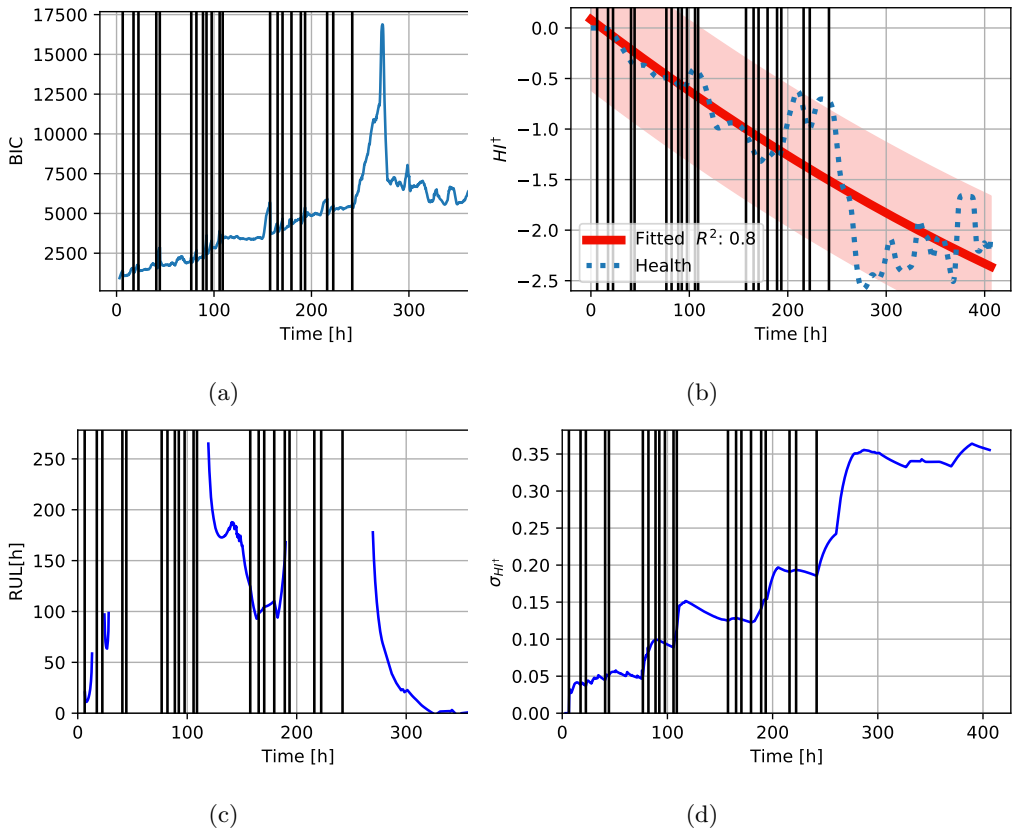


Figure 7.15: Results of mechanical setup. (a) is the BIC evolution (b) is the health index $HI^{\dagger}$, (c) is the online As-HMM RUL predictions, (d) is the standard deviation of the health index regression

Recall that no RTF data (i.e., training data) is used for this analysis, and the health index and RUL are computed as the data arrives at the processor (i.e., the simulated data stream processing). In Fig. 7.15a the evolution of the BIC score is highlighted. Although the model is adapted, a clear increasing trend can be observed in the BIC score. At the end of the process, 21 hidden states were learned. On the other hand, the first concept drift was detected during the first hours of operation. In Fig. 7.15b the health index $HI^{\dagger}$ results are shown. In this case, $HI^{\dagger}$ shows a slow degradation process with some short health recovering phases, which

violates the assumptions in Section. 7.1. When observing the corresponding RUL prediction in Fig. 7.15c, it can be observed that the RUL went from being predictable to unpredictable at different times. This behavior is caused by the observed health recoveries as in the case of Bearing2_2. The health recoveries may cause a change in the sign convexity in the prediction curve; in such cases, the RUL cannot be computed; in spite of that, as observed in the latest RUL predictions, at the time 270 h, the RUL decreases and the regression predicts 170 hours of remaining useful life. As time goes, this prediction converges at 320 h as the time when the failure threshold is overpassed. Although the assumptions made in Section **??** were violated, the RUL predictions and $HI^{\dagger}$ gave insights of a bearing that was evolving towards a failure state. However, the extreme degradation from the beginning provides insights into the bearing installation on the machine or quality issues of the brand-new ball bearing.

Regarding the uncertainty of the prediction for this data-set and mechanical set-up, Fig. 7.15d shows the evolution of the health index regression. In this case, at the first 100 hours of operation, the uncertainty level was around 0.05 orders of magnitude. However, since there was a change in the sign of the concavity of the health index, it caused an important increase in the uncertainty. At 200 hours, a change in the sign of the concavity in the health index caused again an increase in the uncertainty of the health index regression, reaching a maximum of 0.35 orders of magnitude. Notice that with this level of uncertainty, the dispersion of the prediction can be important and affect the RUL prediction as observed in the shaded curves in Fig. 7.15b, where the shaded area is bigger than in the FEMTO data-sets. Therefore, the condition of no health recoveries is important to minimize the RUL uncertainty.

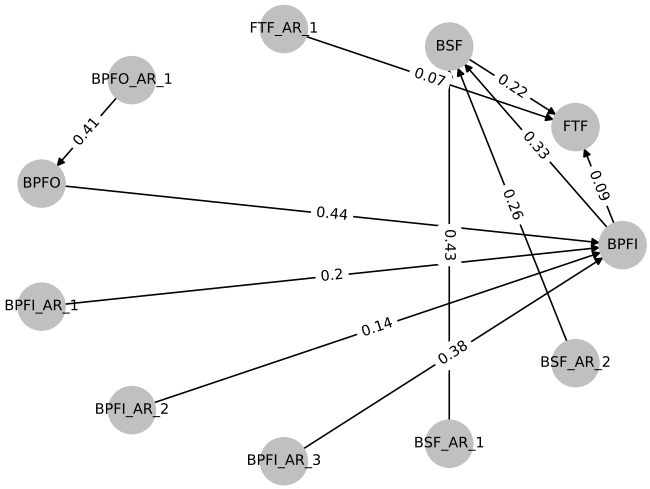


Figure 7.16: Learned Bayesian networks from the own testbench testbed

As previously, the model can estimate the possible probabilistic relationships between variables. This step plays a pertinent role in the interpretation of the ball bearing degradation. In particular, in this case, the resulting Bayesian network of the most degraded observable

state in Fig. 7.16 is shown. A degradation in this ball bearing caused the BSF and FTF frequency-amplitudes to depend on the PBFI frequency-amplitude. The BPFI frequency-amplitude depended on the BPFO frequency-amplitude. In this way, it was interpreted that the outer ring of the ball bearing drove the process, and the bearing rollers and cage depended totally on the behavior of the inner and outer ring. Finally, all the frequencies had a certain degree of dependency on the past, which is the BPFI frequency-amplitude, which required more past values to be explained (3 auto-regressive past values). In contrast, the BPFO frequency-amplitude had only one AR dependency.

7.4 Conclusions

This chapter presented a complete online predictive health assessment solution in terms of an algorithm and its deployment, to monitor ball bearings in real IIoT environments. This machine learning-based solution can work without previous ball bearing RTF data, which is one of the main challenges in the industry. Specifically, the proposed methodology output is the ball bearing health status, which is expressed in different orders of magnitude from a healthy state, and hours for the RUL of the ball bearing.

As natural degradation is expected in real industrial environments, a concept drift detection methodology was used to automatically update an adaptive asymmetric HMM when novel trends appear.

As the new methodology works in embedded devices, from the computational point of view, a performance analysis study of the code has been performed, highlighting that the code is highly dependent on how data structures are stored in the memory of the edge device. Also, the performance of the code is directly correlated with the density of floating-point instructions. The obtained results gave us the fundamental insights to design the algorithm parametrization strategy towards an optimum performance under limited computing power environments.

Different real applications of ball bearings were tested under separate operating conditions to test the proposed methodology, showcasing useful actionable insights that maintenance practitioners can use.

The proposed methodology was compared with other state-of-the-art methodologies. Although every methodology has its own definition of health index, when our health index was compared to other health indexes such as Malahanobis distance from APCMDs, BIC score from WPD or Cauchy-Schwarz divergence from AHMMs, it was observed that our health measure was more informative and interpretable for all the time series and was robust to noise and outliers. In terms of fitness, when compared to a non-adaptative methodology such as WPD, it was observed that the fitness or BIC score of our methodology was more stable and informative. Regarding RUL prediction, it was observed that under certain conditions, it can be predicted; otherwise, the predictions are no longer accurate. Additionally, an operational threshold must be tuned depending on the application, influencing the RUL prediction. In spite of these drawbacks, the ball bearing health reading alone can be a good indicator

of performance. Finally, the asymmetric HMM was capable of learning, for each data series, a set of Bayesian networks that are useful to give further insights into the evolution of the dynamic process, probabilistic dependencies and degradation process.

Chapter 8

A naïve-FSS online methodology

In recent years, the advances in electronics, data processing and storage have enabled industries to perform continuous surveillance of their assets using sensors. In the data processing phase, several mechanical, thermal, electrical and other type of variables are measured. However, some of them may not be always relevant for the underlying dynamic process; therefore, it is crucial to have a model or algorithm capable of determining the relevant variables depending on the current state of the asset. As explained in Section 2.5, ball bearings can be characterized by their fundamental frequencies. Thus, a straightforward question arises: which frequencies and harmonics are relevant and when do they begin to be important?

In spite that several works can be found in FSS in data streams in supervised problems, many fewer appear in unsupervised problems (**?**). A short review of those methodologies was done in Section 3.4.

In this chapter a feature saliency model for online ball bearing FSS is proposed. In particular, Zhu et al. [2012], Adams et al. [2016] and Zheng et al. [2018] developed variants of HMMs, where a set of feature saliencies were used to determine which variables were relevant to describe the data. In Zhu et al. [2012] and Zheng et al. [2018], a variational Bayesian method was used to maximize the log-likelihood of the model and learn the parameters, whereas in Adams et al. [2016] a maximum a posteriori approach was used to learn the model parameters. The previous models were developed only for offline analysis.

In this chapter, a first approximation to solve the problem of FSS online in unsupervised problems is proposed. The unsupervised model of Adams et al. [2016], denoted as FS-HMM, is applied as the cornerstone for a data stream unsupervised FSS methodology with an application to ball bearings surveillance. This model is chosen since it has a simple formulation, interpretation and it is easy to implement. In this application, ball bearing frequencies are recorded and the goal is to determine dynamically their relevancy. As it will be shown, the proposed methodology updates the relevant features when needed and not whenever a new instance or chunk of data arrives as in the previous mentioned articles. This chapter is a short version of the results from Puerto-Santana et al. [2022d] and adapted to match the thesis notation. The chapter is structured as follows: Section 8.1 explains the proposed methodology. Section 8.2 describes the synthetic and real data used for validation.

Then, Section 8.3 shows the obtained results. Finally, Section 8.4 rounds the chapter off with the relevant conclusions.

In Chapter 9 the ideas and concepts proposed in this chapter will be extended, and a more informative model based on As-HMMs is used for FSS in a data stream environment. This is why this chapter is entitled *Naive-FSS*.

8.1 Proposed methodology

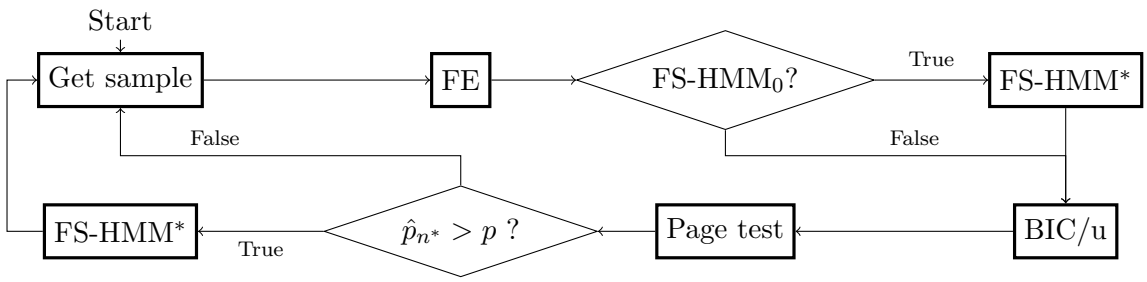


Figure 8.1: Flow diagram of the proposed methodology. The idea is to update the model and the relevant features whenever a novel trend is detected.

Here, the proposed data-stream procedure to update the FS-HMM is introduced. During the model update, the set of relevant features are also updated. Fig. 8.1 shows a flow diagram of the proposed methodology. The first step is to obtain ball bearing frequential data from sensors using the FE algorithm (see Section 2.5. Then, the first FS-HMM* is created if no previous FS-HMM model is available (first iteration). Later, the Bayesian information criterion per unit of observation (BIC/u) is computed using the current FS-HMM. These steps are done with the boxes *FS-HMM** and *BIC/u*. Whereas, the decision node *FS-HMM*$_0$? asks whether there exists a FS-HMM or not. Next, the BIC/u scores are used in the Page sequential test (see Section 2.4.1) to indicate whether the current observations are outliers or not. For this case, given a model $\boldsymbol{\lambda}$, the s page function is:

$$s^t(\boldsymbol{x}^{0:L+t\triangle L}) = \frac{1}{t}\sum_{l=0}^{t}\frac{\mathrm{BIC}(\boldsymbol{x}^{0:L+l\triangle L}|\boldsymbol{\lambda})}{l} \tag{8.1}$$

Where L is an initial data window, and $\triangle L$ is the increment of the data window. If the percentage of outliers overpasses a percentage of permissible outliers in a window given by the Chernoff bounds (see Section 2.4.2), then, a re-training process is performed and the model is updated with the up to time collected data. The Page sequential test is done with the box *Page test* and with the question node $\hat{p}_{n^*} > p$*?* it is determined if updating the model is required or not. The updating process is done in the box *FS-HMM**. Finally, the methodology goes back to the captured data to continue the process of model learning and updating.

8.2 Experimental setup

The number of hidden states of the FS-HMM must be fixed beforehand. For synthetic data, from the construction of the data, as it is shown below, the number of hidden states is three. In the case of real data this is unknown and models with three, four and five hidden states were tried. However, the best temporal BIC/u was obtained using three hidden states, i.e., lower maximum and minimum BIC/u scores were attained with three hidden states. Additionally, the length of the first window for training data was $L = 256$; later, the window was increased by $\triangle L = 10$ data points. The decision rule to detect anomalies with the Page test in this case is:

$$r_t = \begin{cases} 1 & \text{if } s^t(\boldsymbol{x}^{0:L+t\triangle L}) - \min_{l=0,\ldots,t-1} s^l(\boldsymbol{x}^{0:L+l\triangle L}) > \gamma \\ 0 & \text{if } s^t(\boldsymbol{x}^{0:L+t\triangle L}) - \min_{l=0,\ldots,t-1} s^l(\boldsymbol{x}^{0:L+l\triangle L}) \leq \gamma \end{cases} \tag{8.2}$$

Where $\gamma = \ln(\Phi)$, here Φ, as in Section 7.1.2, can be seen as the maximum permissible quotient of likelihood per testing observation over the likelihood per training observation. In this case study $\Phi = 3$. From the Chernoff bounds, the window to determine anomalies was of size $n^* = 87$ and $p = 10\%$.

8.2.1 Synthetic data

It is assumed that there are eight variables which may change their dynamical behavior over time. The goal is to determine when a variable is totally irrelevant or it changes to be relevant. Also, it is assumed that the process is described by normal Gaussian distributions $\mathcal{N}(\mu, \sigma^2)$. The parameters used for this experiment are provided in Table. 8.1.

X_1			X_2			X_3			X_4		
μ	σ	t	μ	σ	t	μ	σ	t	μ	σ	t
-3.0	1.5	$[0, 1500]$	-2.0	2.5	$[0, 1500]$	2.3	2.1	$[0, 1500]$	0.0	1.2	$[0, 3000]$
4.0	1.2	$(1500, 3000]$	6.0	1.2	$(1500, 3000]$	2.8	1.1	$(1500, 8000]$	-2.6	2.5	$(3000, 6000]$
10.0	2.1	$(3000, 6000]$	4.0	1.2	$(3000, 6000]$				3.1	0.8	$(6000, 8000]$
-3.0	1.5	$(6000, 8000]$	-2	2.5	$(6000, 8000]$						

X_5			X_6			X_7			X_8		
μ	σ	t	μ	σ	t	μ	σ	t	μ	σ	t
-3.2	1.5	$[0, 8000]$	2.7	2.5	$[0, 8000]$	-5.3	1.4	$[0, 8000]$	1.6	1.2	$[0, 1500]$
									7.8	1.0	$(1500, 3000]$
									-6.5	2.0	$(3000, 6000]$
									1.6	1.2	$(6000, 8000]$

Table 8.1: Parameters used to generate the synthetic data

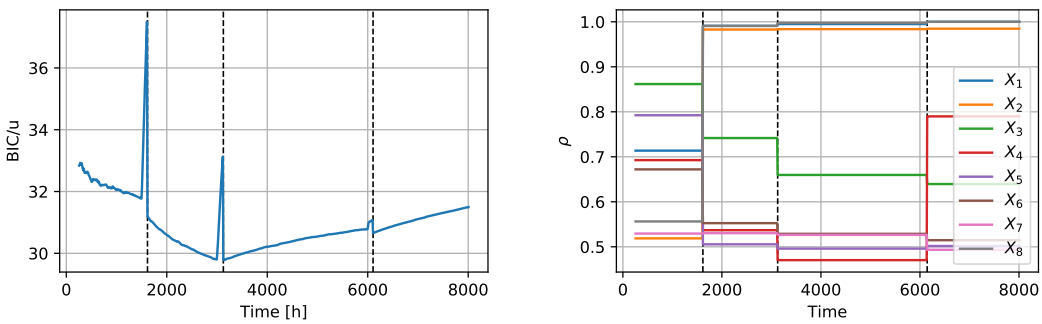
The parameters are chosen such that variables X_1, X_2 and X_8 are relevant variables. On the other hand, variables X_5, X_6 and X_7 are irrelevant since they are Gaussian noise. Variable X_4 is at first sight irrelevant but becomes relevant with the progression of periods. Variable X_3 becomes irrelevant over time.

8.2.2 Real data

The purpose of this ball bearing testing set up is to monitor vibrations over time during ball bearing useful life. In this case, the set-up and signals from the Aingura testbed from Section 7.2 are used. Recall that inside the CMAI, the FE procedure is performed and the fundamental frequency amplitudes and up to four harmonics are computed and stored (20 variables). An Eco 6004-2RS ball bearing is tested at a radial force of 2.41 kN at 3180 RPM (53 Hz). The ball bearing runs during T=400 hours or equivalently 2.5 times its theoretical operation life.

8.3 Results

8.3.1 Synthetic data



(a) (b)

Figure 8.2: Evolution of the relevancy level for the different variables. Whenever a dotted vertical line is observed, it implies that an updating model procedure was done.

In Fig. 8.2 the results obtained from the synthetic data are drawn. Vertical dotted lines are used to indicate novel concept drift detection. In (a) the BIC/u evolution is shown. Note that, after an important growth in BIC/u, the model was re-trained and the BIC/u trend decreased and the model fitness becomes stable. In particular, three updating processes were performed at times close to $t = 1500, 3000, 6000$ related to the variable definitions in Table. 8.1. The delay in the updating process was caused by the Chernoff bounds; however, these Chernoff bounds are necessary to prevent unnecessary model updating due to outliers. In (b) the evolution of relevancies of the variables X_1 to X_8 are drawn. Before the first novel detection, only one feature had a relevancy ρ greater than 0.9, this is reasonable since before $t = 1500$ all the features behaved like noise. After $t = 1500$ variables X_1, X_2 and X_8 had a high relevancy level (close to 1); however variables X_3, X_4, X_5, X_6 and X_7 had a low relevancy value. Nevertheless, the variable X_4 obtained more relevancy with each re-updating process since its dynamical behavior becomes more clear. On the other hand, variables X_5, X_6 and X_7 which were built as noise, kept a low level of relevancy after all the re-updating processes. From this experiment, it can be determined that the model can discriminate between noise

and relevant features dynamically, and the model was updated only when needed.

8.3.2 Real data

Fig. 8.3 (a) shows the evolution of the amplitude of the fundamental frequencies. Note that, the FTF amplitude is the frequency which shows the greater values and changes. In comparison, the BPFI and BPFO amplitudes show the lowest values and the dynamical changes are less evident.

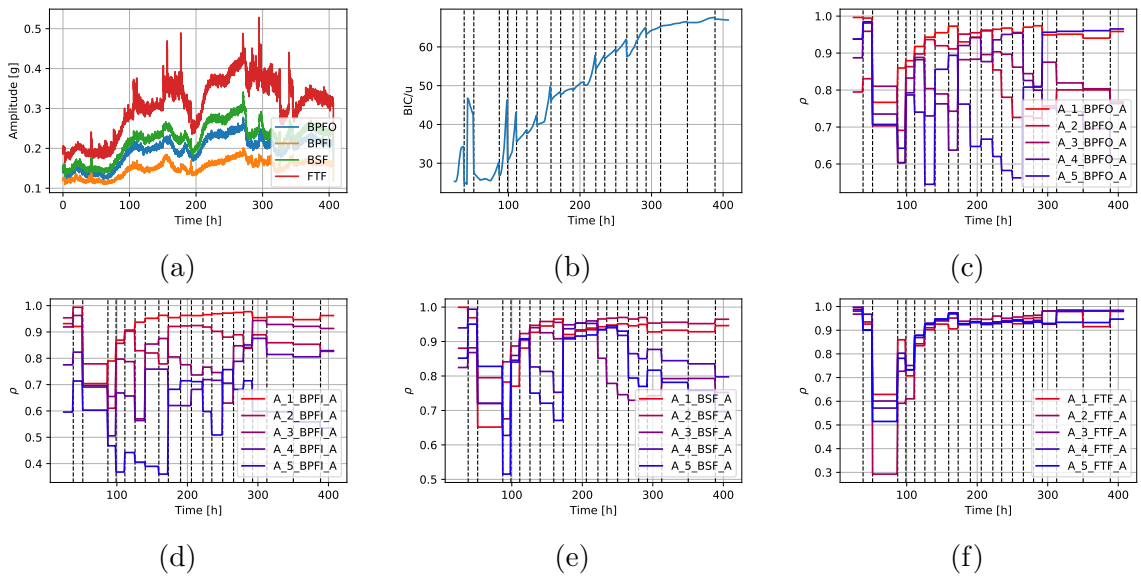


Figure 8.3: (a) recorded fundamental frequencies amplitudes. (b) evolution of BIC/u. (c), (d), (e) and (f) show the progressions of the relevancy levels for different features. Dotted vertical lines imply that an updating model procedure was done.

Observe in Fig. 8.3 (b) that the BIC/u score was always going up in spite of the several novel concept drifts detected. However, at the end of the process, the data was stable and fewer re-updates were needed.

Set $A\_i\_f\_A$ as the $(i-1)$-harmonic of the f frequency (being the 0-harmonic the fundamental f frequency). (c) corresponds to the progression of the relevancy for the BPFO and its harmonics. It is noticeable that the level of relevancy of the fundamental frequency was high for the whole process, whereas its harmonics were less relevant. In particular, the fourth harmonic showed the greatest decay in relevancy to become irrelevant. As for the BPFO case, the fundamental frequency for BPFI in (d) was always relevant, whereas its harmonics were less important In particular, the first harmonic showed low to intermediate levels of relevancy, but at later periods it increased its relevancy. In the case of BSF in (e), the fundamental frequency and its first harmonic were the most relevant features. It is also remarkable that the remaining harmonics had times of high or low relevancy. In (f), the harmonics of FTF sometimes were more relevant than the fundamental frequency during the ball bearing evolution.

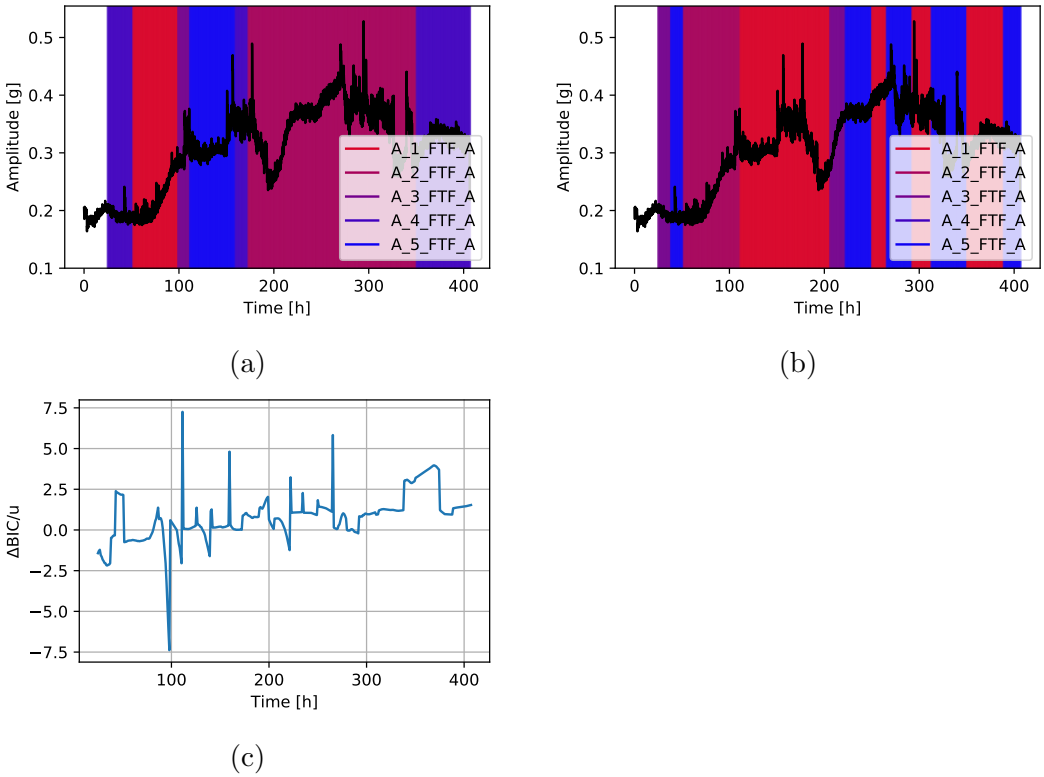


Figure 8.4: FTF amplitude segmented by (a) the most relevant feature and (b) the least relevant feature. (c) shows the temporal difference ΔBIC/u

Since the FTF was the fundamental frequency with the more evident dynamic changes, a deeper analysis of this feature is carried out. In Fig. 8.4 (a) the FTF amplitude progression was segmented by the most relevant harmonic. In particular, during the first important change at time $t = 60$, the fundamental frequency was the most relevant feature; however, after that, it did not appear again. On the other hand, the first harmonic was the most relevant feature for a large time period at $t \in [180, 360]$. In (b), the evolution of the least relevant harmonic is shown; in particular, the first harmonic, was the least relevant at the early stages of the data stream to later become the most relevant harmonic. Also, it is remarkable that there were times where the fundamental frequency was the least relevant; which implies that information would be lost if the harmonics are omitted from the analysis. Therefore, some harmonics played a relevant role in describing the dynamical data and the role changes over time. Accordingly, monitoring frequency/harmonics is of crucial interest.

Finally, in (c) wit was computed $\Delta\text{BIC/u} = (\text{BIC/u})_{\ddot{\imath}} - (\text{BIC/u})_{\text{FS}}$ where $(\text{BIC/u})_{\text{FS}}$ is the BIC/u by using a FS-HMM model and $(\text{BIC/u})_{\ddot{\imath}}$ using a naïve-HMM **?**. The mean of (c) was $\overline{\Delta\text{BIC/u}} = 0.74 > 0$, i.e., $\overline{(\text{BIC/u})_{\ddot{\imath}}} > \overline{(\text{BIC/u})_{\text{FS}}}$ which suggests that for this study, the use of FS-HMM improved the performance.

8.4 Conclusions

This chapter adapted an offline unsupervised machine learning model to an online dynamic FSS methodology based on novel concept drift detection in data streams. To demonstrate its applicability within real-world scenarios, the approach has been used to determine relevant frequency amplitudes of a ball bearing. The methodology was capable of changing the subset of relevant features when needed in both synthetic and real data instead of computing the relevancy whenever a new instance or chunk of data arrived, as it was done in previous methodologies (Huang et al. [2015], Shao et al. [2016], Fahy and Yang [2019a]). Moreover, the algorithm could capture the evolution of relevancy for fundamental frequencies and their harmonics. For real data and depending on the ball bearing part, for its corresponding frequency amplitude, some harmonics may be more or equally relevant as the fundamental frequency; also, this relevancy could change over time. Nevertheless, the change in relevancy was only obtained when a novel drift is detected and not when a concept drift arise. Also, since the FS-HMM is being used, all the variables were assumed to be independent which, as seen before in Chapters 4, is not always the case for the whole ball bearing operational life. These issues are addressed in the next chapter, where a new kind of asymmetric FS-HMM is introduced, which is capable of detecting feature drifts. Finally, no complexity or computational study was performed in this case since this methodology was designed as a proof of concept to determine the viability to use embedded feature selection algorithms in data streams.

Chapter 9

Online FSS with As-HMMs

FSS is an issue that has been largely reviewed and studied in the literature. The benefits from performing a correct feature selection are well-known, such as, improving model learning and prediction, model simplification, accelerating the computational speed, reducing data storage and others. Nevertheless, when working with data-streams, selecting features is a harder issue: relevant and non-redundant features may change over time. With the development and popularity of paradigms such as the 4IR or the internet of things, the urge to extract and detect relevant information in data streams is increasing. Hence, this chapter tries to face this problem and provides a solution.

Recalling from Section 2.4, three kinds of concept drift in the data stream distribution can occur: real drift, virtual drift and feature drift. The proposed model in this chapter, and its updating scheme for data streams, deal mainly with virtual and feature drifts. The real drift issue is acknowledged as a consequence of a virtual or feature drift: whenever a virtual or feature drift arises, a new distribution on the class/cluster variable given those features appears (real drift).

The chapter is structured as follows: Section 9.1 explains the proposed model with its learning algorithm and its adaption to work on online data-stream environments. Section 9.2 describes the synthetic and real data used for validation, and the corresponding results. Finally, Section 9.3 rounds the chapter off with the relevant conclusions and future work.

9.1 Model proposal

In this chapter a batch and online locally feature saliency asymmetric HMM is introduced. The model of this chapter is an extension of the model proposed in Chapter 6 where the feature saliencies depend on the hidden state. In spite that the batch learning algorithm based on the EM algorithm is presented here for the sake of theoretic completeness, the focus will be on an online version for IIoT applications. In the case of the data stream, the localized feature saliencies will let us identify feature drifts, and the relevant features will be able to change with the hidden state.

For the proposed algorithm, the Viterbi and forward-backward algorithm from Chapter 5 can be used to perform partially the E-step and estimate the most likely sequences of hidden states and also the sequence of relevancies or feature drifts.

9.1.1 Local feature saliency asymmetric HMMs for batch analysis

In this contribution it is assume that the emission probabilities are a mixture of Gaussian noise and AR-AsLG-HMM. Thus, depending on the hidden state, the Bayesian network and the relevant features may change. This highly flexible model will be referred to as LFS-AsHMM.

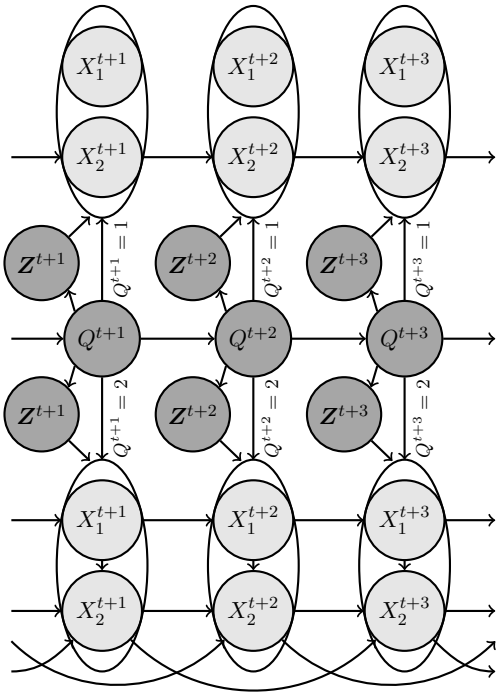


Figure 9.1: Example of an LFS-AsHMM pictured as a dynamic Bayesian network

The embedded FSS process assumes that irrelevant features are not affected by changes in hidden states, therefore a Bernoulli vector $\boldsymbol{Z}^t = (Z_1^t, ..., Z_M^t)$ is introduced in the model with probability:

$$\zeta_i(\boldsymbol{z}^t) := P(\boldsymbol{z}^t | Q^t = i, \boldsymbol{\lambda}) = \prod_{m=1}^{M} \rho_{im}^{z_m^t} (1 - \rho_{im})^{(1 - z_m^t)}, \tag{9.1}$$

where $\rho_{im} := P(Z_m^t = 1 | Q^t = i, \boldsymbol{\lambda})$ for $m = 1, ..., M$. Note that is is assumed that the Z_m^t Bernoulli variables are conditional independent between them and that the ρ_{im} parameters change with the hidden state. On the other hand, the irrelevant behavior is modeled for each variable with a Gaussian distribution with parameters ϵ_m for the mean and τ_m^2 for the

variance. The dependency of $\boldsymbol{X}^t$ given $\boldsymbol{Z}^t$ and p^* AR past values is modeled as:

$$b_i(\boldsymbol{x}^t|\boldsymbol{z}^t) := P(\boldsymbol{x}^t|\boldsymbol{x}^{t-p^*:t-1}, \boldsymbol{z}^t, Q^t = i, \boldsymbol{\lambda}) = \prod_{m=1}^{M} f_{im}(x_m^t)^{z_m^t} g_m(x_m^t)^{(1-z_m^t)}, \tag{9.2}$$

Where $f_{im}(x_m^t) = \mathcal{N}(x_m^t|\boldsymbol{u}_{im}^t\boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t\boldsymbol{\eta}_{im}, \sigma_{im}^2)$ is the probability density function for the relevant component, whereas $g_m(x_m^t) = \mathcal{N}(x_m^t|\epsilon_m, \tau_m^2)$ is the noise term. Also, $\boldsymbol{u}_{im}^t = (1, u_{im1}^t, ..., u_{imk_{im}}^t)$ and $\boldsymbol{d}_{im}^t = (x_m^{t-1}, ..., x_m^{t-p_{im}})$ are vectors with the values of the k_{im} parents of X_m^t in the Bayesian network graph and its $p_{im} \leq p^*$ past values. Whereas $\boldsymbol{\beta}_{im} = [\beta_{im0}, ..., \beta_{imk_{im}}]^\top$ and $\boldsymbol{\eta}_{im} = [\eta_{im1}, ..., \eta_{imp_{im}}]^\top$ are their corresponding. weights.

Observe in Fig. 9.1 an example of the new model topology, with two variables/features. When $Q^t = 1$, no probabilistic relationships appear between X_1^t and X_2^t, also X_2^t depends on one AR value or (X_2^{t-1}). When $Q^t = 2$, there is a probabilistic dependency of (X_2^t) from (X_1^t), additionally, X_1^t depends on one AR value. X_1^{t-1} and X_2^t depends on two AR values (X_2^{t-1} and X_2^{t-2}). Finally, $\boldsymbol{X}^t$ on both contexts, $Q^t = 1$ and $Q^t = 2$, depends on the binary vector $\boldsymbol{Z}^t$.

From Eq. (9.1) and Eq. (9.2) the emission probabilities can be derived:

$$b_i(\boldsymbol{x}^t) := P(\boldsymbol{x}^t|\boldsymbol{x}^{t-p^*:t-1}, Q^t = i, \boldsymbol{\lambda}) = \prod_{m=1}^{M} \rho_{im} f_{im}(x_m^t) + (1 - \rho_{im}) g_m(x_m^t), \tag{9.3}$$

and the full information probability can be written as follows:

$$P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) = \pi_{q^{p^*}} \prod_{t=p^*}^{T-1} a_{q^t q^{t+1}} \prod_{t=p^*}^{T} \zeta_{q^t}(\boldsymbol{z}^t) b_{q^t}(\boldsymbol{x}^t|\boldsymbol{z}^t). \tag{9.4}$$

9.1.1.1 E-step

As in previous chapters, the first step to derive the EM updating equations is to define the auxiliary function. Given a current model $\boldsymbol{\lambda}^{(s)}$, the auxiliary function is defined as:

$$\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) := \sum_{R(\boldsymbol{Q}^{p^*:T})} \sum_{R(\boldsymbol{Z}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}, \boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \tag{9.5}$$

From Eq. (9.5),the log-likelihood (LL) from Section 5.2 for the $\boldsymbol{\lambda}$ model can be computed:

$$\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \mathcal{H}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + \ln P(\boldsymbol{x}^{p^*:T}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) = \mathcal{H}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) + LL(\boldsymbol{\lambda}), \tag{9.6}$$

where

$$\mathcal{H}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{R(\boldsymbol{Q}^{p^*:T})} \sum_{R(\boldsymbol{Z}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}^{(s)}) \ln P(\boldsymbol{q}^{p^*:T}, \boldsymbol{z}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}). \tag{9.7}$$

By Eq. (9.6), Eq. (9.7) and Section 5.2, it is known that each iteration of the EM algorithm with $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ implies improvements in the likelihood function.

Introducing Eq. (9.4) in Eq. (9.5) a tractable expression of $\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})$ is obtained. It will be useful to find the updating formulas of the model parameters:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = &\sum_{i=1}^{N}\gamma^{p^*}(i)\ln(\pi_i^{p^*}) + \sum_{t=p^*}^{T-1}\sum_{i=1}^{N}\sum_{j=1}^{N}\xi^t(i,j)\ln(a_{ij}) \\ &+ \sum_{t=p^*}^{T}\sum_{i=1}^{N}\sum_{m=1}^{M}\psi_m^t(i)\ln(\rho_{im}f_{im}(x_m^t)) + \sum_{t=p^*}^{T}\sum_{i=1}^{N}\sum_{m=1}^{M}\phi_m^t(i)\ln((1-\rho_{im})g_m(x_m^t)).\end{aligned} \tag{9.8}$$

In Eq. (9.8), the latent a posteriori probabilities are:

$$\begin{aligned}&\gamma^t(i) := P(Q^t = i|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}), && \xi^t(i,j) := P(Q^{t+1} = j, Q^t = i|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}),\\ &\psi_m^t(i) := P(Q^t = i, Z_m^t = 1|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}), && \phi_m^t(i) := P(Q^t = i, Z_m^t = 0|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)}),\end{aligned} \tag{9.9}$$

for $t = p^*, ..., T$, $i = 1, ..., N$ and $m = 1, ..., M$. The E-step consists of estimating these quantities. In the case of $\psi_m^t(i)$:

$$\begin{aligned}\psi_m^t(i) &= P(Q^t = i, Z_m^t = 1|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)})\\ &= P(Z_m^t = 1|Q^t = i,\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)})P(Q^t = i|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}^{(s)})\\ &= \frac{P(x_m^t, Z_m^t = 1|\boldsymbol{x}_m^{t-p^*:t-1}, Q^t = i,\boldsymbol{\lambda}^{(s)})\gamma^t(i)}{P(x_m^t|\boldsymbol{x}_m^{t-p^*:t-1}, Q^t = i,\boldsymbol{\lambda}^{(s)})}\\ &= \frac{\rho_{im}f_{im}(x_m^t)\gamma^t(i)}{\rho_{im}f_{im}(x_m^t) + (1-\rho_{im})g_m(x_m^t)}.\end{aligned} \tag{9.10}$$

It is not hard to note that $\gamma^t(i) = \phi_m^t(i) + \psi_m^t(i)$ for $m = 1, ..., M$ and $i = 1, ..., N$. Therefore $\phi_m^t(i) = \gamma^t(i) - \psi_m^t(i)$ and:

$$\phi_m^t(i) = \frac{(1-\rho_{im})g_m(x_m^t)\gamma^t(i)}{\rho_{im}f_{im}(x_m^t) + (1-\rho_{im})g_m(x_m^t)}. \tag{9.11}$$

Now, $\gamma^t(i)$ is estimated as follows:

$$\gamma^t(i) = P(Q^t = i|\boldsymbol{x}^{0:T},\boldsymbol{\lambda}') = \frac{\alpha_{p^*}^t(i)\beta_{p^*}^t(i)}{\sum_{j=1}^{N}\alpha_{p^*}^t(j)\beta_{p^*}^t(j)}. \tag{9.12}$$

In the previous equation the forward variable is $\alpha_{p^*}^t(i) := P(Q^t = i, \boldsymbol{x}^{p^*:t}|\boldsymbol{x}^{0:p^*-1},\boldsymbol{\lambda}^{(s)})$ and the backward variable is $\beta_{p^*}^t(i) := P(\boldsymbol{x}^{t+1:T}|Q^t = i,\boldsymbol{x}^{0:t},\boldsymbol{\lambda}^{(s)})$. The forward-backward algorithm stated in Section 5.3 must be applied to estimate $\alpha_{p^*}^t(i)$ and $\beta_{p^*}^t(i)$. Finally, $\xi^t(i,j)$ can be computed as:

$$\xi^t(i,j) = \frac{\alpha_{p^*}^t(i)a_{ij}b_j(\boldsymbol{x}^{t+1})\beta_{p^*}^{t+1}(j)}{\sum_{u=1}^{N}\sum_{v=1}^{N}\alpha_{p^*}^t(u)a_{uv}b_v(\boldsymbol{x}^{t+1})\beta_{p^*}^{t+1}(v)}. \tag{9.13}$$

9.1.1.2 M-step

The M-step corresponds to optimizing Eq. (9.8) with respect to the model parameters. The following theorem gives the updating formulas that result from the optimization.

Theorem 9.1. Assume there is a current model $\boldsymbol{\lambda}^{(s)}$ from which the E-step has been computed in the formulas Eq. (9.9). By maximizing Eq. (9.8), the resulting parameter $\boldsymbol{\lambda}^{(s+1)}$ can be obtained with the following updating formulas.

The feature saliencies $\{\rho_{im}^{(s+1)}\}_{m=1}^{M}$ are updated as:

$$\rho_{im}^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_m^t(i)}{\sum_{t=p^*}^{T} \gamma^t(i)}. \tag{9.14}$$

The initial distribution $\boldsymbol{\pi}^{(s+1)} = \{\pi_i^{(s+1)}\}_{i=0}^{N}$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{9.15}$$

The transition matrix $\boldsymbol{A}^{(s+1)} = \{a_{ij}^{(s+1)}\}_{i,j=1}^{N}$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{9.16}$$

The mean and variance, $\{\epsilon_m^{(s+1)}\}_{m=1}^{M}$ and $\{(\tau_m^2)^{(s+1)}\}_{m=1}^{M}$, from the noise component, are updated as:

$$\begin{aligned} \epsilon_m^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i) x_m^t}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)} \\ (\tau_m^2)^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)(x_m^t - \epsilon_m)^2}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)}. \end{aligned} \tag{9.17}$$

Denoting $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ for $m = 1, ..., M$, $t = p^*, ..., T$ and hidden state $i = 1, ..., N$, the relevance parameters $\boldsymbol{\eta}_{im} = \{\eta_{imr}\}_{r=1}^{p_{im}}$, $\boldsymbol{\beta}_{im} = \{\beta_{imk}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear equation system:

$$\begin{cases} \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t = \sum_{t=p^*}^{T} \psi_m^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{9.18}$$

if $\boldsymbol{\theta}_{im} = [\boldsymbol{\beta}_{im} | \boldsymbol{\eta}_{im}]^\top$, $\boldsymbol{o}_{im}^t = [\boldsymbol{u}_{im}^t | \boldsymbol{d}_{im}^t]$, and $\boldsymbol{\Psi}_{im}^{p^*:T} := \text{Matrix}([\psi_m^{p^*}(i), ..., \psi_m^T(i)])$. Then, the

previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{9.19}$$

Setting $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$; then, $\{(\sigma_{im}^2)^{(s+1)}\}_{i,m=1}^{N,M}$ can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_m^t(i)(x_m^t - \hat{\nu}_{im}^t)^2}{\sum_{t=p^*}^{T} \psi^t(i)}. \tag{9.20}$$

The proof of this theorem is provided in the Appendix D. Observe that the updating equations of the M step for this algorithm are the same as the ones from the FS-AsHMM (see Section 6.2.2), but ρ_{im} parameters are computed differently. Regarding the computational complexity of the linear equations in Eq. (9.18), the same comments made in Section 5.4 are valid for this case.

9.1.1.3 Structural EM (SEM)

In this chapter the greedy-forward algorithm proposed in Section 5.6 is used to search the space of possible graphical models. Also, the noise definition given in Section 2.3.6 is used to describe relevancy. Therefore, a restriction is imposed during the search of structures such that no noise variable is added to any context-specific Bayesian network. The restriction consists of omitting any possible arc coming into or from variables X_m which fulfill the following condition: $\rho_{im} \leq \bar{\rho}$, where $\bar{\rho} \in [0, 1)$ is a threshold that determines which variables are relevant. Recall that the opposite of this assumption is not true, i.e., if a variable does not have any relationship with any other variable in a context-specific Bayesian network, it does not mean that it is irrelevant or noise under our relevance definition.

9.1.2 Localized feature saliencies asymmetric HMM for online analysis

In our first approximation for FSS in online environments, the methodology introduced in Chapter 8 performed feature drifts whenever a novel concept was discovered. Such strategy, also assumed that all variables were independent. In this section, an online learning mechanism is provided to use the LFS-HMMs in data-streams to perform FSS and overcome the previous issues.

It will be assumed here that at the beginning of the data stream, there is only one concept and therefore only one hidden state will be learned. The idea is to increase the number of hidden states whenever a novel concept is detected in the data stream. As in the methodology in Chapter 7, the number of hidden states will evolve over time. Also, due to the flexibility of context-specific feature saliencies, the feature saliencies learned for each hidden states are not lost and the relevancy of each feature can be tracked during the online analysis. Additionally, the online learning phase must be adjusted to meet the previous requirements.

9.1.2.1 Online processing

The online processing consists of an updating scheme whenever a concept drift is detected. The updating scheme that is presented here is a modification of the model update proposed in Section 7.1.1. However, since local feature saliencies are added to the model, the online scheme must be adapted.

At the beginning of any data-stream, a LFS-AsHMM with only one hidden state is learned from L instances and its BIC/u score per unit data is saved for the Page test. The window of data is increased in $\triangle L$ data units. Here the Page-test and Chernoff bounds are applied as in Section 8.1 to detect novel trends. If a novel concept appears, a new hidden state is added to the model and the SEM algorithm is executed only for the parameters related to the new added hidden state.

Suppose that the current model $\boldsymbol{\lambda} = \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}\}$ has N hidden states. Assume that at the instance $\boldsymbol{x}^t$ a concept drift is detected and the model is updated. A new prior model $\boldsymbol{\lambda}' = \{\boldsymbol{A}', \boldsymbol{B}', \boldsymbol{\pi}'\}$ is used to update the known information form the data stream.

To add a new hidden state in the transition matrix, the augmented matrix $\boldsymbol{C}$ is introduced:

$$\boldsymbol{C} = \begin{bmatrix} & & & | & y_0 \\ & \boldsymbol{A} & & | & \vdots \\ & & & | & y_0 \\ - & - & - & - & - \\ \frac{1}{N+1} & \cdots & \frac{1}{N+1} & | & \frac{1}{N+1} \end{bmatrix}, \tag{9.21}$$

where y_0 must be a positive small number, in this study $y_0 = 1 \times 10^{-6}$. $\boldsymbol{C}$ will be used to generate a new prior transition matrix $\boldsymbol{A}'$ which enables the new model to determine the probabilistic transitions between the new observed hidden state and the previous learned hidden states. Set $a'_{ij} = \frac{c_{ij}}{\sum_{j=1}^{N+1} c_{ij}}$. To address the problem that the new learned matrix $\boldsymbol{A}'$ may not be a Markov chain matrix, the rows of $\boldsymbol{A}$ are normalized.

For the prior values of $\boldsymbol{\eta}_{N+1,m}$, $\boldsymbol{\beta}_{N+1,m}$ and $\sigma^2_{N+1,m}$, it is assumed that the new hidden state in its relevant component can be represented as a naïve Bayes network with no AR components. More precisely:

$$\begin{aligned} f_{N+1,m}(\boldsymbol{x}^t) &= \mathcal{N}(\boldsymbol{x}^t | \beta_{N+1,m0}, \sigma^2_{N+1,m}) \\ \beta_{N+1,m0} &= \frac{1}{L}\sum_{i=1}^{L} x_m^{t-i}, \\ \sigma^2_{N+1,m} &= \frac{1}{L}\sum_{i=1}^{L} (x_m^{t-i} - \beta_{N+1,m})^2, \end{aligned} \tag{9.22}$$

It is expected that if a variable is actually noise to a process, then its parameters will be fixed for all the data stream. Therefore, the parameters ϵ_m and τ_m^2 are set for all the data stream

as follows:

$$\epsilon_m = \sum_{i=1}^{L} \frac{x_m^i}{L}, \quad \tau_m^2 = \sum_{i=1}^{L} \frac{(x_m^i - \epsilon_m)^2}{L}. \tag{9.23}$$

These parameters are never updated. Finally, the parameter $\boldsymbol{\pi}'$ is updated as follows:

$$\boldsymbol{\pi}' = [\boldsymbol{\pi}|0] \tag{9.24}$$

Or the concatenation with a zero to right. Once the prior model $\boldsymbol{\lambda}'$ is generated, the SEM algorithm is executed only on the parameters $\{A_{j,N+1}\}_{j=1}^{N}$, $\{A_{N+1,j}\}_{j=1}^{N}$, $\boldsymbol{\beta}_{N+1}$, $\boldsymbol{\eta}_{N+1}$ and $\boldsymbol{\sigma}_{N+1}^2$.

In many cases, the data streams may seem "infinite" and it is not possible to store all the captured data; hence a maximum window size $L_{\max}$ is imposed such that the learning and inference process do not require more than $L_{\max}$ instances. Once a new instance arrives and the buffer of size $L_{\max}$ is already filled, the oldest instance is forgotten and the new instance is added to the processing buffer.

9.2 Experimental setup

The experiments are focused on an online analysis, where the model must learn the parameters as the data arrives and determine the feature relevancies.

It is well known that a linear Gaussian Bayesian network can be expressed as a multivariate Gaussian distribution Koller and Friedman [2009]. Let $\boldsymbol{\mu}_i = \{\mu_{im}\}_{m=1}^{M}$ be the mean vector corresponding to the i-linear Gaussian Bayesian network in the context $Q = i$. For these experiments, the labeling g equation (see Section 5.7) is defined as:

$$g(i) = \sum_{m=1}^{M} |\mu_{im}| \chi_{\{\rho_{im} \geq \bar{\rho}\}} + \sum_{m=1}^{M} |\epsilon_m| \chi_{\{\rho_{im} < \bar{\rho}\}}, \tag{9.25}$$

where χ_δ is the indicator function. It is one if the condition δ is met and zero otherwise. This g function adds the true information from all the features, the greater the g value, the greater the magnitude of the observed data.

The proposed methodology from this section is compared with the strategies DFM-MCFS proposed in Fahy and Yang [2019b] and FSMCP exposed in Ma et al. [2020] (see Section 3.2). These methodologies are selected since they are the most recent ones regarding feature selection in data streams in unsupervised problems. In the first case, the methodology updates the relevant features whenever a buffer of data is filled. Additionally, as stated in Fahy and Yang [2019b], the methodology is compatible with any clustering model. Hence, the Gaussian mixture model is used, which has been previously used for online clustering Diaz-Rozo et al. [2020]. In the second case, the features are updated whenever a new data point arrives. For the proposed methodology in this chapter, the model and the feature relevancies are updated when needed.

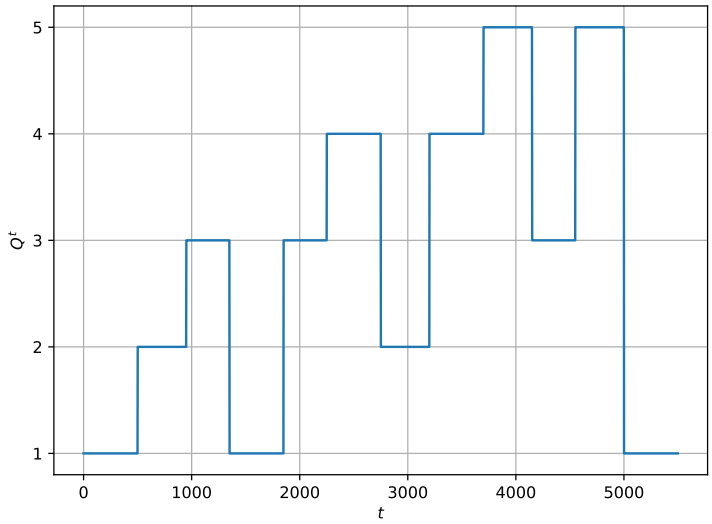


Figure 9.2: Sequence of hidden states to generate synthetic data

9.2.1 Synthetic data

9.2.1.1 Description

The synthetic data consists of $M = 10$ variables. Seven are relevant, two are noise, namely X_6 and X_9, one X_3 is initially noise but it becomes relevant for certain time instances. It is assumed that the data contain $N = 5$ hidden states. The parameters of the synthetic data re exposed in the Appendix D.

To simulate the data, a sequence of hidden states is required. For this study, the sequence pictured in Fig. 9.2 is used. Two datasets are generated from this sequence, a time series for training and another for testing. All the models do an online training phase and then, their current parameters are tested. The learned feature relevancies are discussed. In the case of the proposed online methodology, the $g(i)$ evolution in the data stream is exhibited and some of the learned context-specific Bayesian networks are drawn to gain insights. The value of the initial window size is $L = 128$ with a maximum of $L_{max} = 4096$, and every 10 new instances, the BIC score is computed. Additionally $r\bar{h}o = 0.9$, is used to select the features that are relevant and are used to created context-specific Bayesian networks.

9.2.1.2 Results

Fig. 9.3 shows the results in the testing phase of LFS-AsHMM. From the training data, six hidden states were discovered or inferred from the data. During the testing phase, one additional hidden states was found. The time where the hidden state was found is marked with a vertical dotted line. Note from the figure that variables X_6 and X_9 had a low relevancy as expected, since they behaved as noise during all the dynamical process (they have a constant value of 1). Variable X_3 had some moments of high relevancy due that its parameters differed from the noise level for some hidden states. The remaining variables showed an

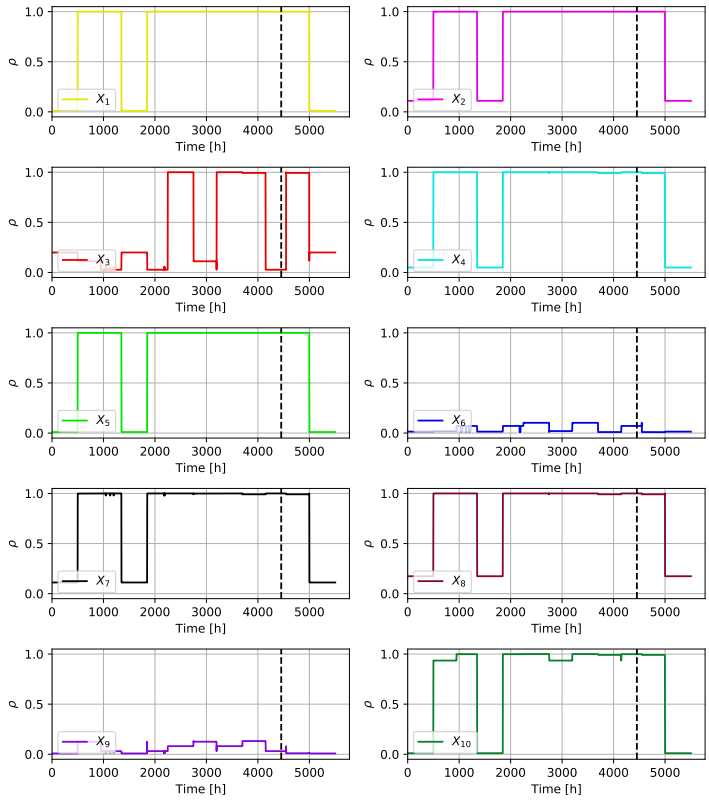


Figure 9.3: Feature saliencies and drifts discovered during the testing phase by the proposed model LFS-AsHMM. The changes in feature relevancies are computed using the Viterbi algorithm.

expected behavior, since their relevancies were close to one most of the time. Nonetheless, it is relevant to observe that for some time instances, the relevancy of all variables went close to zero. The reason behind this behavior is that the initial state (where all the variables are considered as noise) was again present in the data-stream. This implies that the noise component represented better the data and the relevancies in such cases had to be close to zero.

Table 9.1 shows the feature saliencies (ρ_{im}) during the training phase (with 6 hidden states) and the testing phase (where $Q = 7$ is added). Notice that indeed, X_6 and X_9 were noise variables, since for all the discovered hidden states, their relevancy was close to zero. Variable X_3 was irrelevant for most of the hidden states, but relevant in two of them $Q = 5, 6$. The remaining variables had a high relevant value for most of the hidden states as expected.

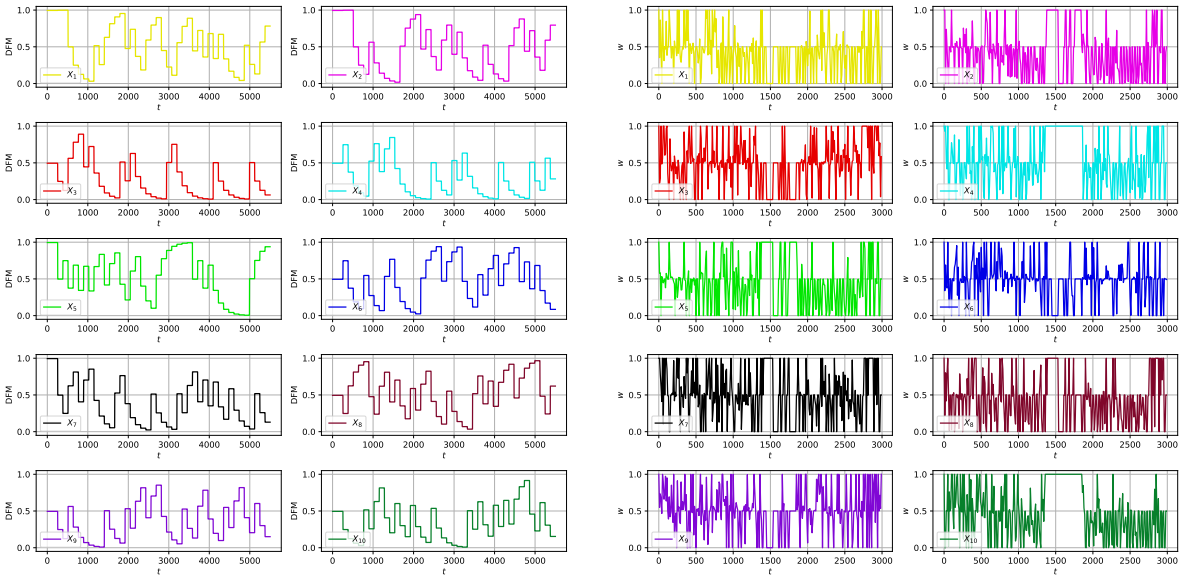
Recall that it is assumed that at the first hidden state $Q = 1$, all the variables are noise, since no other concept is known. However, whenever a novel concept is discovered and the model is updated, the features may change their status.

In Fig. 9.4 (a), the results of DFM-MCFS are presented. Observe that the relevancies had a greater variance than in the case of four proposed methodology. In particular, variables X_6 and X_9 had time periods where their relevancy was high, which is not true by the construction of the dataset. Furthermore, relevant variables such as X_4 and X_{10} had a low relevancy for all

$Q \setminus$ M	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
1	0.01	0.11	0.2	0.05	0.01	0.01	0.11	0.17	0.01	0.01
2	1.0	1.0	0.11	1.0	1.0	0.02	1.0	1.0	0.13	0.94
3	1.0	1.0	0.05	1.0	1.0	0.01	0.98	1.0	0.01	1.0
4	1.0	1.0	0.03	1.0	1.0	0.07	1.0	1.0	0.03	1.0
5	1.0	1.0	1.0	1.0	1.0	0.1	1.0	1.0	0.08	1.0
6	1.0	1.0	0.99	0.99	1.0	0.01	0.99	0.99	0.01	0.99
7	0.51	1.0	0.12	0.51	1.0	0.01	1.0	1.0	0.01	0.51

Table 9.1: Feature saliencies discovered for all the discovered hidden states during the learning and testing phase

the dynamical process, which is again, incorrect. Additionally, it is observed that the features changed their relevancy whenever the buffer of data was filled. In our case, the relevancy was unchanged unless an actual concept drift in data appeared.

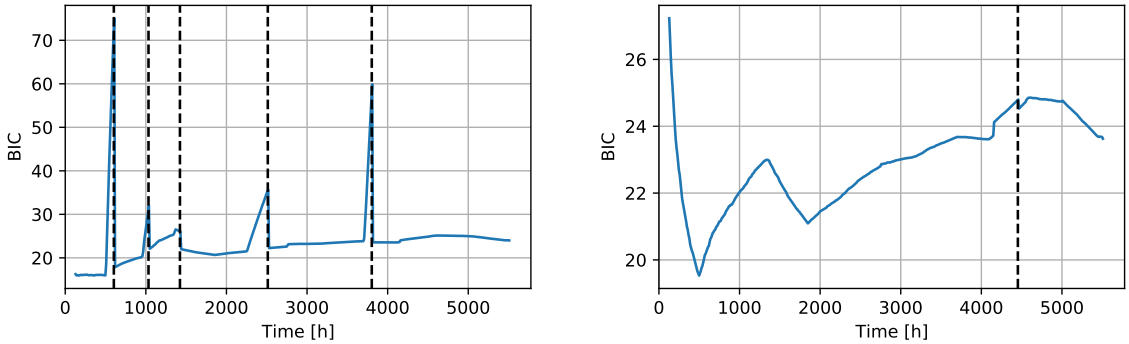


(a) DFM-MCFS (b) FSMPC

Figure 9.4: Feature drifts discovered during the testing phase by (a) the DFM-MCFS and (b) FSMPC models

In Fig. 9.4 (b), the results of FSMPC are pictured. Observe that the relevancies had an even higher variance over the time. Relevant features such as X_4, X_5 and X_{10} had time instants where their relevancies were high as expected. However, there were also times where they were incorrectly low. Regarding the noisy variables X_6 and X_9, their relevancy level was highly fluctuanting. There were time points where it was one or close to one which is incorrect. Recall that this methodology updates the feature relevancies whenever a new instance arrives which explains the high variability. Nevertheless, the results were not as expected.

To contrast the learning and testing phases of the proposed algorithm, in Fig. 9.5 the BIC score per unit data is plotted. In the training phase, the BIC score per unit of data had time periods where it increased abruptly until a novel concept appeared and a model retraining



(a) Training (b) Testing

Figure 9.5: BIC per unit data comparison between training and testing phases

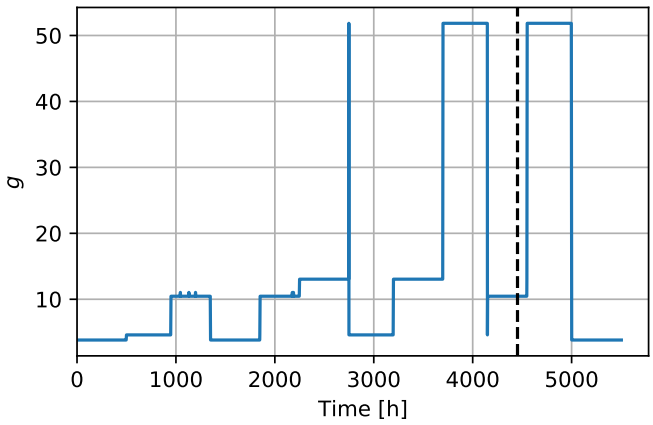


Figure 9.6: Sequence of hidden states inferred during the testing phase thanks to the g function

was performed Such events are highlighted with vertical dotted black lines. In the testing data, the BIC score per unit data was more stable and only one novel concept was found. This implies that the learned model in the training phase was not representative enough to explain all the testing phase.

Regarding the model inference, Fig. 9.6 includes the online clustering and its interpretation given by the Viterbi algorithm and the $g(i)$ function. Observe that, if the Viterbi path is compared to the real hidden state sequence plotted in Fig. 9.2, the prediction fits fairly the ground truth with some outliers at the state transitions. In this sense, it is observed that the model can be useful to learn and discover the intrinsic states and features drifts in data.

Recall that this is a methodology for unsupervised data, and therefore, the models must discover patterns in data. Proof of this is the difference between the true number of hidden states (five from the data description) and the estimated number of hidden states (seven in this case); but, in spite of that, the model could provide data insights which were helpful to understand and analyze the time series.

9.2.2 Real data

9.2.2.1 Ball bearing degradation

Ball bearings are relevant mechanical components inside industrial and non-industrial machines. It is known that due to the application of mechanical loads such as high forces or temperatures, ball bearings suffer degradation and breakage. In industrial applications, the damage of one ball bearing can compromise the productivity of manufacturing lines and induce losses in time and money Larrañaga et al. [2018].

For this application, the dataset described in Section 4.3.1 is used. As in Chapter 6, harmonics are used for the analysis, however, in this chapter, it is online estimated the relevancy of each feature for each time instance and not in a global offline manner. Nevertheless, at early stages of the ball bearing life, harmonics and fundamental frequencies can be seen as noise and be irrelevant for the ball bearing health estimation. In this study, the aim is to determine dynamically the features that are relevant and those which are not.

For this application, again, the focus will be the ball bearing B3, since it fails in S1 and S3. The signals S1 and S2 will be used to train the online model, which will be used to analyze the signal S3. In the training signals, ball bearing B3 failed in its inner race. In the testing signal, ball bearing B3 failed due to its outer race.

The fundamental frequencies and three harmonics as variables are used, hence sixteen features are used. The learned baseline model from the learning phase, will be used in the testing phase to detect feature drifts. In the case of a novel concept due to an unknown concept or feature drift, it will be added to the model.

9.2.2.2 Ball bearings results

Table 9.2 shows the obtained relevancies during the training phase. The variables with index 1, refer to the fundamental frequency; index i refers to the $i-1$ harmonic $i > 1$. the most relevant result is that at the latest discovered trend, all the frequencies are relevant as expected. Nonetheless, for certain discovered trends, specially $Q = 2$, it is observed that some variables do not overpass the threshold $\rho_{im} > \overline{\rho}$ as FTF_1 and BSF_1, $BPFO_3$, BSF_3, $BPFI_4$ and BSF_4. However, at the end of the process, the relevancies tended to increase with some exceptions.

In the online testing phase, the recorded evolution of relevancies is observed in Fig 9.7. For most of the ball bearing life, all the features behaved as noise. Nonetheless, some hours before the ball bearing breakage ($t = 1000h$), all the features drifted to a relevant level, with some exception, where noisy drifts were observed, namely in BSF_1, $BPFO_2$, BSF_3, FTF_2 and $BPFI_3$ the relevacies oscillated between different levels. In spite of that noisy end behavior, the model was capable of detecting the moment where the model was running to failure. It is noticeable that some relevancies were not used (see for example, BSF_1 relevancies at $Q = 2$). This may indicate that in the training phase, more concepts were observed during the ball bearing breaking process, and such were not observed in the testing phase. Additionally, it is remarkable that in the testing phase, no novel trends were detected and the model never

$Q\backslash$ m	$BPFO_1$	$BPFI_1$	BSF_1	FTF_1	$BPFO_2$	$BPFI_2$	BSF_2	FTF_2
1	0.2	0.04	0.01	0.02	0.02	0.21	0.24	0.03
2	0.99	0.97	0.44	0.86	1.0	1.0	1.0	1.0
3	1.0	0.94	1.0	0.66	0.99	1.0	1.0	1.0
4	1.0	1.0	0.85	0.98	0.85	1.0	1.0	0.76
5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97
$Q\backslash$ m	$BPFO_3$	$BPFI_3$	BSF_3	FTF_3	$BPFO_4$	$BPFI_4$	BSF_4	FTF_4
1	0.21	0.24	0.16	0.05	0.03	0.15	0.26	0.03
2	0.84	0.91	0.26	1.0	1.0	0.86	0.81	1.0
3	1.0	1.0	0.94	0.97	1.0	1.0	0.85	0.01
4	1.0	0.89	0.67	1.0	1.0	0.97	1.0	0.99
5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Table 9.2: Feature relevancies ρ_{im} from the proposed model for the different frequency magnitudes found in the B3 learning and testing phases

had to self update. In this sense, the learned model could be representative enough for the testing data.

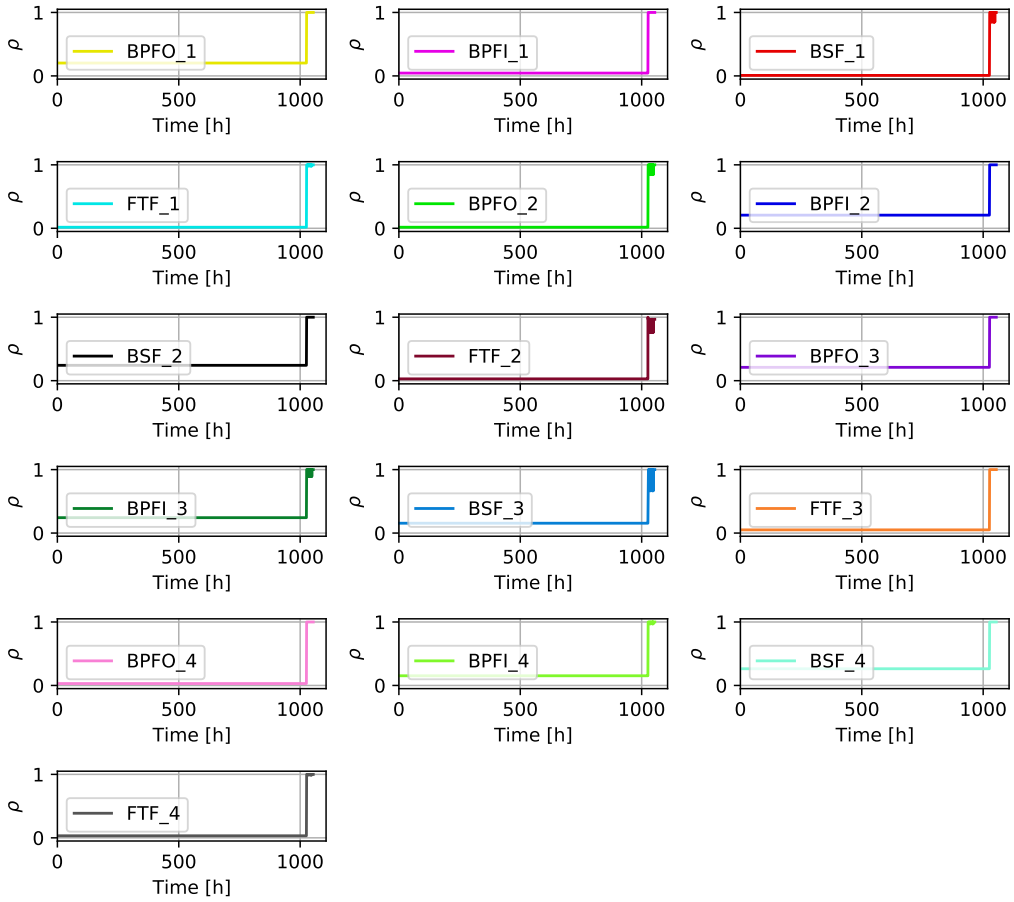
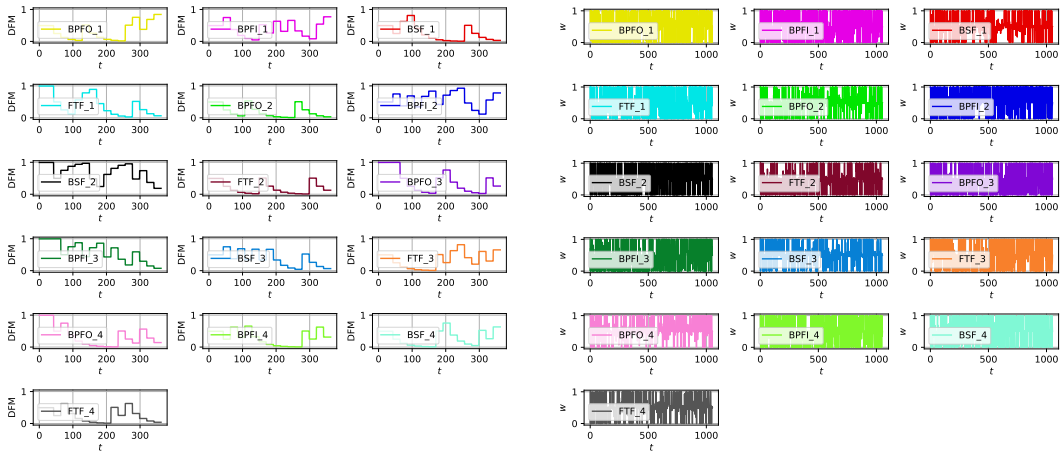


Figure 9.7: Feature drifts discovered during the testing phase by the our model LFS-AsHMM

Fig 9.8 shows the relevancy results for the other two methodologies. In (a), the FDM-MCFS methodology plots a periodic behavior in all feature relevancies. However, none of them reached an important level of relevancy for all the ball bearing life as it was observed



(a) DFM-MCFS (b) FSMPC

Figure 9.8: Feature drifts discovered during the testing phase by (a) the DFM-MCFS and (b) FSMPC models

with our model. Nonetheless, the low relevancies were also latent during the ball bearing failure times, which is not desired since the methodology could not provide insights into the ball bearing degradation. In (b), the FSMPC strategy shows a noisy behavior for all the ball bearing life which is not helpful, since it did not provide any kind of feature insight or process understanding.

Fig 9.9 exhibits the evolution of hidden states (Viterbi path with the g function) during the testing phase. This sequence shows that the summed magnitudes of the ball bearing were statistically constant during most of the ball bearing life. However, as in the case of the feature relevancies, the magnitudes increase drastically in the ball bearing final hours, indicating, as before, a worsening of the ball bearing health.

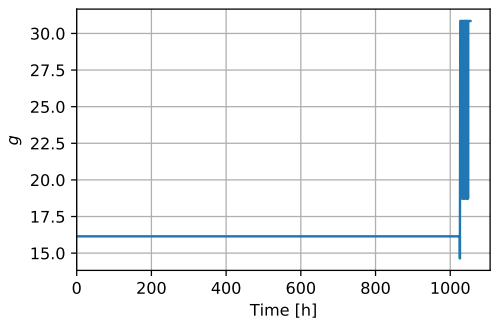
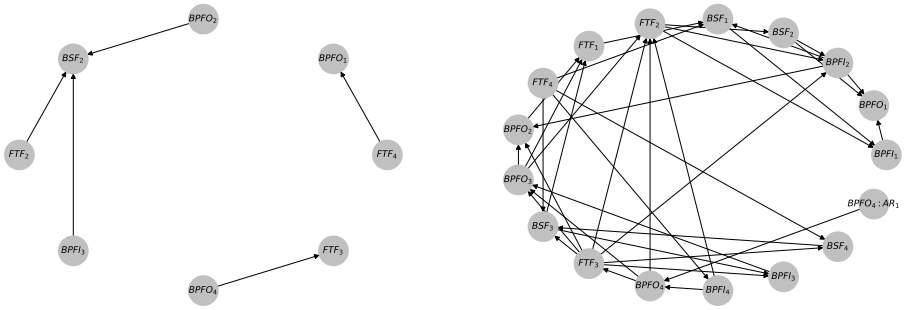


Figure 9.9: Sequence of hidden states inferred by LFS-AsHMM during the testing phase of the ball bearing

Finally, as stated in the introduction, the model is capable of learning context-specific graphical models. These models can be used as an explanatory tool to understand and give further data insights. Recall that the space of possible graphical models is determined by



(a) (b)

Figure 9.10: A couple of context-specific Bayesian networks learned by LFS-AsHMM from the degradation of the ball bearing

the features which fulfill the condition $\rho_{im} > \overline{\rho}$. Fig. 9.10 (a), is a graph when the ball bearing was not in an advanced degradation state in the training phase. Observe that the BPFO and the FTF harmonic and fundamental frequencies were related, or in other words, the behavior of the ball bearing outer ring and jail were dynamically dependent. Also, note that the BSF first harmonic depended on harmonics of the BPFI, BPFO and FTF, which indicated that the ball bearing rollers depended on the behavior of the other ball bearing components. The graph of Fig. 9.10 (b) corresponds with a state where the ball bearing was heavily damaged in the training phase. In this scenario, several dependencies between the ball bearing frequencies are observed, an evidence that the dynamical system was completely coupled and all the ball bearing parts were relevant for the degradation process. Additionally, an autoregressive behavior in the third harmonic of the BPFO ($BPFO_4 : AR_1$) is observed, which means that also the past values of the ball bearing were relevant to explain the current level of degradation.

In short, the proposed methodology does not only detect relevant features drifts but also provides data insights. In addition, it is indeed capable of detecting the number of hidden states or concepts of a process, and it is capable of self updating in testing data if required.

9.3 Conclusions

In this chapter, a new methodology to detect and learn feature drifts in continuous unsupervised data was proposed. The proposed methodology used a new embedded feature selection algorithm based on As-HMMs. The model has the property of "local feature relevancy", which enables it to change the relevancy level of the features depending on the hidden states/concepts discovered in data. The methodology allows the model to self determine its number of hidden states as the data arrives in a data stream. Once a model is learned, it can be used in testing data to detect concept drifts using the Viterbi algorithm. In each concept a set of relevant features is selected and a context-specific Bayesian network is used to explain the data. These context-specific Bayesian networks with the localized feature relevancies, can

provide deeper data understanding. Finally, a hidden state labelling function was proposed to determine the magnitude of changes in concepts or hidden states taking into account the relevant and irrelevant features of each context.

The model was validated using synthetic and real data to degradation of ball bearings. It was compared to other techniques of the state of the art which determine dynamically the relevance of features in unsupervised problems. It was observed that the proposed methodology obtained the most consistent results, and not only could provide feature relevancies dynamically but also useful data insights. In spite of the previous results, the proposed model is still not embedded into the CMAI module. As future work, it is pertinent to determine the time cost of the LFS-AsHMM in order to ensure data integrity and feasible processing times.

Chapter 10

Kernel Density estimation on As-HMMs

In real data, hypotheses such as homoscedasticity (constant variance) or independence between input variables do not hold, and traditional models for continuous data, such as AR or moving average (MA) linear regression (or combinations of these), are often unsuitable. As seen in this thesis, dynamic probabilistic graphical models such as HMMs can alleviate such issues and be used to provide further data insights, compute likelihoods, perform data segmentation or classification. However, traditional and the extended HMMs, are usually based on the Gaussian distribution with linear functions in the mean which limits their modeling capabilities to Gaussian data. To overcome this issue, MoG-HMMs have been proposed; nonetheless, due to numerical issues as underflow/overflow when computing inverses of covariance matrices, such models have to assume diagonal covariance matrices, i.e., independence among the model features. More recently, as exposed in Chapter 4, As-HMMs have been proposed to model variable dependencies in a more stable computational manner and using fewer parameters than in the case of MoG-HMMs, via the use of context-specific Bayesian networks. However, such models also assume Gaussian data with linear functions in the mean.

In this chapter, we propose a new kind of As-HMMs, where non-Gaussian or non linear in mean dynamic data is modeled using KDE estimations. By adding context-specific dependencies between variables to the model. These new models are referred to kernel density estimation in asymmetric hidden Markov models or KDE-AsHMMs.

The new model is validated with synthetic and real data from ambient sound and from CNC drilling machines. The model is compared in terms of classification accuracy and log-likelihood to previous HMM-based models. Additionally, theoretical bounds on computational time complexity of the learning and inference (log-likelihood computation) algorithms are provided.

In short, this chapter provides the following contributions to the state of the art related to HMMs:

1. A new kind of asymmetric HMMs is introduced to model non-linear data based on KDEs and context-specific Bayesian networks.

2. All the parameters are interpretable and can provide further data insights.

3. A learning algorithm, based on the EM and structural EM algorithms, is given to learn KDE-AsHMMs.

4. Complexity bounds on computation time are derived for further model understanding.

The chapter is organized as follows. Section 10.1 presents KDE-AsHMMs, their learning algorithm and theoretical time complexity upper bounds. Section 10.2 describes the validation and relevant comparisons with synthetic and real data. Finally, Section 10.3 rounds the article off with the conclusions and considerations based on the model validation findings.

10.1 Model proposal

10.1.1 Definition

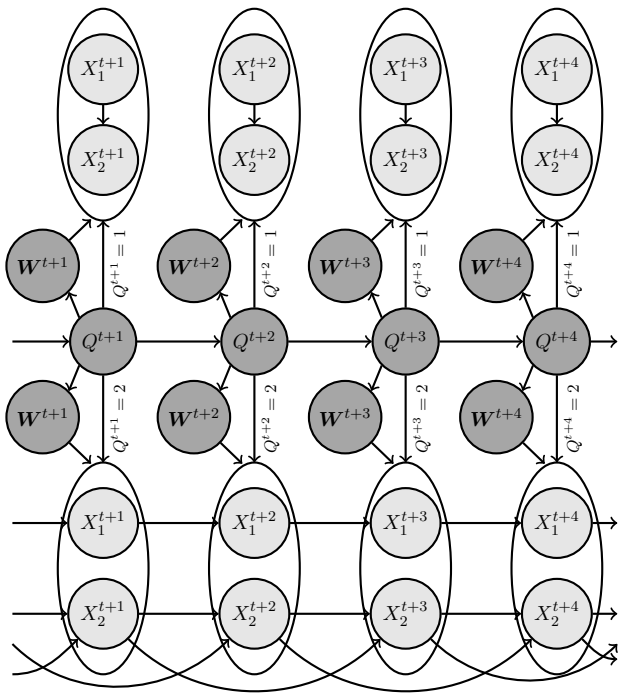


Figure 10.1: Example of a KDE-AsHMM pictured as a dynamic Bayesian networks

The core idea behind this new model is to describe and identify non-Gaussian and non-stationary dynamic processes by combining HMMs and KDEs. The proposed model, denoted KDE-AsHMM, can be described as a dynamic Bayesian networks, see Figure 10.1. In the figure, the added latent variables $\boldsymbol{W}^t$ follow a categorical distribution, which depends on the latent variables $\boldsymbol{Q}^t$. The latent variable $\boldsymbol{W}^t$ is used to determine the most representative

instances to be used in the kernel density estimation for each hidden state. The observable variables $\boldsymbol{X}^t$, can depend statistically on each other through a directed acyclic graph structure, as well autoregressive dependences on their most recent prior values, up to a maximum order $P*$. This value p^* can be chosen by the experimenter or calculated using the Yule-Walker equations, see Box and Jenkins [1976]. Nevertheless, these relationships may change depending on the hidden state as pictured in Figure 10.1: when $Q^t = 1$, X_2^t depends on X_1^t, but when $Q^t = 2$, the relationship changes and X_1^t instead depends on X_1^{t-1}, whereas X_2^t depends on X_2^{t-1} and X_2^{t-2}. In this manner, the complexity of the emission probability distributions can increase as the data requires.

The different conditional dependencies provided by Figure 10.1 are defined as follows. Assume that N hidden states, M variables are being modeled and the length of the training data is $L+1$ and the length of the test data is $T+1$. Regarding the transition probabilities:

$$P(Q^t = q^t | Q^{t-1} = q^{t-1}; \boldsymbol{\lambda}) = a_{q^{t-1}q^t}, \text{ with } \sum_{j=1}^{N} a_{q^{t-1}j} = 1. \tag{10.1}$$

With respect to the latent variable $\boldsymbol{W}^t = \{W_{p^*}^t, W_{p^*+1}^t, ..., W_L^t\}$, which determines the relevant training samples for each hidden state, its conditional probabilities are defined as:

$$P(\boldsymbol{W}^t | Q^t; \boldsymbol{\lambda}) = \prod_{l=p^*}^{L} (\omega_{Q^t,l})^{W_l^t}, \quad \sum_{l=p^*}^{L} \omega_{Q^t l} = 1, \quad \sum_{l=p^*}^{L} W_l^t = 1, \quad W_l^t \in \{0,1\}. \tag{10.2}$$

Assume that the training data is $\boldsymbol{y}^{0:L}$, during the training phase $T = L$, and, as will be argued below, it is necessary that $W_l^l = 0$ for $l = 0, ..., L$; in this manner, issues with degenerate likelihoods and infinitely narrow kernels are avoided, see Piccardi and Perez [2007].

Let $\text{Set}(\boldsymbol{X}^t) := \{X_1^t, ..., X_M^t\}$ the set of features of the model. We denote the $\kappa_{q^t m}$ parents of X_m^t at $q^t \in R(Q^t)$ as $\{V_{q^t mk}^t\}_{k=1}^{\kappa_{q^t m}} \subset \text{Set}(\boldsymbol{X}^t)$. With this notation, the conditional densities of the observable variables in this article are defined as:

$$f_{\boldsymbol{X}^t|\boldsymbol{W}^t,Q^t}(\boldsymbol{X}^t|\boldsymbol{W}^t, Q^t; \boldsymbol{\lambda}) = \prod_{m=1}^{M} f_{X_m^t|\boldsymbol{W}^t,Q^t,\boldsymbol{U}_{Q^t,m}^t}(X_m^t|\boldsymbol{W}^t, Q^t, \boldsymbol{U}_{Q^t,m}^t; \boldsymbol{\lambda}), \tag{10.3}$$

where each $\boldsymbol{U}_{Q^t m}^t = (V_{Q^t m1}^t, ..., V_{im\kappa_{Q^t m}}^t, X_m^{t-1}, ..., X_m^{t-p_{Q^t m}})$ is a context-specific random vector, which contains the $\kappa_{Q^t m}$ parents and $p_{Q^t m}$ AR dependencies of the variables X_m^t. In order to have an interpretable and well defined model, for every $q^t \in R(Q^t)$, the set of tuples $\bigcup_{m=1}^{M} \boldsymbol{U}_{q^t m}^t \times \{X_m\}$, must describe the edges of an directed acyclic graph (DAG), i.e., a context-specific Bayesian networks[1].

To analytically describe, the dependencies in the context-specific Bayesian networks, define $\boldsymbol{v}_{Q^t m}^l$ as the instantiation of the random vector $\boldsymbol{U}_{Q^t,m}^t$ by $\boldsymbol{y}^{0:L}$, and define $\boldsymbol{M}_{Q^t,m}$ as a context-specific matrix of size $(\kappa_{Q^t m} + p_{Q^t m}) \times (\kappa_{Q^t m} + p_{Q^t m})$ which determines the weights

[1]Here, $\times$ represents the Cartesian product

of the kernel dependencies, and set:

$$\mu^t_{l,Q^t,m} := y^l_m + \boldsymbol{M}_{Q^t,m}(\boldsymbol{U}^t_{Q^t,m} - \boldsymbol{v}^l_{Q^t,m})^\top. \tag{10.4}$$

We rewrite each factor in Eq. (10.3) in terms of a kernel function K, bandwidth $h_{Q^t,m}$ and centers $\mu^t_{l,Q^t,m}$:

$$f_{X^t_m|\boldsymbol{W}^t,Q^t,\boldsymbol{U}^t_{Q^t,m}}(X^t_m|\boldsymbol{W}^t, Q^t, \boldsymbol{U}^t_{Q^t,m};\boldsymbol{\lambda}) := \prod_{l=p^*}^{L}\left(\frac{1}{h_{Q^t,m}}K\left(\frac{X^t_m - \mu^t_{l,Q^t,m}}{h_{Q^t,m}}\right)\right)^{W^t_l}. \tag{10.5}$$

The previous equations omit the dependencies on $\boldsymbol{y}^{0:L}$ and $\boldsymbol{X}^{t-p^*:t-1}$ to simplify notation. Note additionally that the bandwidths are allowed to vary for each variable for each hidden state. In this manner, the deviations on parents and AR values can be used to correct the kernel as the data requires. Notice that every component of the KDE for each hidden state can be obtained using Eq. (10.2)–(10.4) as follows:

$$f_{\boldsymbol{X}^t,W^t_l=1|Q^t=i}(\boldsymbol{X}^t, W^t_l = 1|Q^t = i;\boldsymbol{\lambda}) = \omega_{il}\prod_{m=1}^{M}\frac{1}{h_{im}}K\left(\frac{X^t_m - \mu^t_{lim}}{h_{im}}\right). \tag{10.6}$$

This is useful to determine the emission probabilities for this model:

$$\begin{aligned} b_i(\boldsymbol{X}^t) &= \sum_{R(\boldsymbol{W}^t)} f_{\boldsymbol{X}^t,\boldsymbol{W}^t|Q^t=i}(\boldsymbol{X}^t, \boldsymbol{W}^t|Q^t = i;\boldsymbol{\lambda}) \\ &= \sum_{l=p^*}^{T} f_{\boldsymbol{X}^t,W^t_l=1|Q^t=i}(\boldsymbol{X}^t, W^t_l = 1|Q^t = i;\boldsymbol{\lambda}) \\ &= \sum_{l=p^*}^{T}\omega_{il}\prod_{m=1}^{M}\frac{1}{h_{im}}K\left(\frac{X^t_m - \mu^t_{lim}}{h_{im}}\right). \end{aligned} \tag{10.7}$$

The full information would be:

$$\begin{aligned} &f_{\boldsymbol{X}^{p^*:T},\boldsymbol{Q}^{p^*:T},\boldsymbol{W}^{p^*:T}}(\boldsymbol{x}^{p^*:T}, \boldsymbol{q}^{p^*:T}, \boldsymbol{w}^{p^*:T};\boldsymbol{\lambda}) = \\ &\pi_{q^{p^*}}\prod_{t=p^*}^{T-1}a_{q^t,q^{t+1}}\prod_{t=p^*}^{T}\prod_{l=p^*}^{L}\left(\omega_{q^t,l}\prod_{m=1}^{M}\frac{1}{h_{q^t,m}}K\left(\frac{X^t_m - \mu^t_{l,q^t,m}}{h_{q^t,m}}\right)\right)^{w^t_l} \end{aligned} \tag{10.8}$$

In log form

$$\begin{aligned} \ln(f) = &\ln(\pi_{q^{p^*}}) + \sum_{t=p^*}^{T-1}\ln(a_{q^t,q^{t+1}}) + \\ &\sum_{t=p^*}^{T}\sum_{l=p^*}^{T} w^t_l\left(\ln(\omega_{q^t,l}) + \sum_{m=1}^{M}\ln\left(\frac{1}{h_{q^t,m}}K\left(\frac{X^t_m - \mu^t_{l,q^t,m}}{h_{q^t,m}}\right)\right)\right) \end{aligned} \tag{10.9}$$

As summary, the proposed model $\boldsymbol{\lambda}$ consists of the set of parameters $\boldsymbol{\lambda} := \{\boldsymbol{\pi}, \mathbf{A}, \boldsymbol{h} := \{h_{im}\}_{i=1,m=1}^{N,M}, \boldsymbol{\omega} := \{\omega_{il}\}_{i=1,l=p^*}^{N,L}, \boldsymbol{M} := \{\boldsymbol{M}_{im}\}_{i=1,m=1}^{N,M}\}$, where $\boldsymbol{M}_{im}$ represents the κ_{im} dependencies of X_m from other variables and its $p_{im} \leq p^*$ AR dependencies. From the indexing, the dependencies can vary with X_m and the hidden state $i \in R(Q^t)$. Also, the dependencies must follow a Bayesian network, and therefore DAG structures must be employed.

10.1.2 Learning algorithm

For this model, the EM algorithm will be applied to learn the models parameters, assume that a model $\boldsymbol{\lambda}^{(s)}$ has been computed or provided. The auxiliary function for this model is:

$$\begin{aligned}&\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \\ &E_{P(\boldsymbol{Q}^{p^*:T},\boldsymbol{W}^{p^*:T}|\boldsymbol{X}^{0:T};\boldsymbol{\lambda}^{(s)})}[\ln f_{\boldsymbol{X}^{p^*:T},\boldsymbol{Q}^{p^*:T},\boldsymbol{W}^{p^*:T}}(\boldsymbol{x}^{p^*:T}, \boldsymbol{Q}^{p^*:T}, \boldsymbol{W}^{p^*:T}; \boldsymbol{\lambda})]\end{aligned} \tag{10.10}$$

Recall that for the training phase $\boldsymbol{x}^{0:T} = \boldsymbol{y}^{0:L}$ and $W_l^l = 0$, and thence Eq. (10.10) can be expressed analytically as:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \sum_{i=1}^{N} \gamma^0(i) \ln(\pi_i) + \sum_{t=p^*}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \zeta^t(i,j) \ln(a_{ij}) + \\ \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \sum_{i=1}^{N} \psi_l^t(i) \left(\ln(\omega_{il}) + \sum_{m=1}^{M} \ln\left(\frac{1}{h_{im}} K\left(\frac{x_m^t - \mu_{lim}^t}{h_{im}} \right) \right) \right)\end{aligned} \tag{10.11}$$

From the previous equation, the following latent variables appear:

$$\begin{aligned}\gamma^t(i) &= P(Q^t = i|\boldsymbol{x}^{0:T}; \boldsymbol{\lambda}^{(s)}) \\ \zeta^t(i,j) &= P(Q^t = i, Q^{t+1} = j|\boldsymbol{x}^{0:T}; \boldsymbol{\lambda}^{(s)}) \\ \psi_l^t(i) &= P(Q^t = i, W_l^t = 1|\boldsymbol{x}^{0:T}; \boldsymbol{\lambda}^{(s)})\end{aligned} \tag{10.12}$$

In particular, $\psi_l^t(i)$ can be computed as follows:

$$\begin{aligned}\psi_l^t(i) &= P(W_l^t = 1|Q^t = i, \boldsymbol{x}^{0:T}; \boldsymbol{\lambda}^{(s)}) P(Q^t = i|\boldsymbol{x}^{0:T}; \boldsymbol{\lambda}^{(s)}) \\ &= P(W_l^t = 1|Q^t = i, \boldsymbol{x}^t; \boldsymbol{\lambda}^{(s)}) \gamma^t(i) \\ &= \frac{P(W_l^t = 1, \boldsymbol{x}^t|Q^t = i; \boldsymbol{\lambda}^{(s)})}{\sum_{k \neq t}^{T} P(W_k^t = 1, \boldsymbol{x}^t|Q^t = i; \boldsymbol{\lambda}^{(s)})} \gamma^t(i) \\ &= \frac{\omega_{il} \prod_{m=1}^{M} K\left(\frac{x_m^t - \mu_{lim}^t}{h_{im}}\right)}{\sum_{k \neq t}^{T} \omega_{ik} \prod_{m=1}^{M} K\left(\frac{x_m^t - \mu_{kim}^t}{h_{im}}\right)} \gamma^t(i).\end{aligned} \tag{10.13}$$

On the other hand, $\gamma^t(i)$ and $\zeta^t(i,j)$ can be computed using the well known forward-backward algorithm. Let us assume that we have a Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \tag{10.14}$$

Set $\overline{\boldsymbol{u}}^t_{iml} := (\boldsymbol{u}^t_{im} - \boldsymbol{v}^l_{im})$ and $\overline{x}^t_{ml} := (x^t_m - y^l_m)$. For the M step, the updating formulas for the parameters $\boldsymbol{M}_{im}$ are deduced as:

$$\frac{\partial \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})}{\partial \boldsymbol{M}_{im}} = \sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \frac{\partial}{\partial \boldsymbol{M}_{im}} \left(\ln \left(K \left(\frac{x^t_m - \mu^t_{lim}}{h_{im}} \right) \right) \right) = 0 \tag{10.15}$$

$$\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \frac{\partial}{\partial \boldsymbol{M}_{im}} \left(-\frac{\left(\overline{x}^t_{ml} - \boldsymbol{M}_{im}(\overline{\boldsymbol{u}}^t_{iml})^\top\right)^2}{2h^2_{im}} \right) = 0 \tag{10.16}$$

$$\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \left((\overline{\boldsymbol{u}}^t_{iml})^\top \left((\overline{x}^t_{ml} - \boldsymbol{M}_{im}(\overline{\boldsymbol{u}}^t_{iml})^\top \right)^\top \right) = 0 \tag{10.17}$$

$$\left(\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) (\overline{\boldsymbol{u}}^t_{iml})^\top \overline{\boldsymbol{u}}^t_{iml} \right) \boldsymbol{M}^\top_{im} = \sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) (\overline{\boldsymbol{u}}^t_{iml})^\top \overline{x}^t_{ml} \tag{10.18}$$

$$(\boldsymbol{M}^\top_{im})^{(s+1)} = \left(\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) (\overline{\boldsymbol{u}}^t_{iml})^\top \overline{\boldsymbol{u}}^t_{iml} \right)^{-1} \left(\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) (\overline{\boldsymbol{u}}^t_{iml})^\top \overline{x}^t_{ml} \right) \tag{10.19}$$

In practice, the linear system in Eq. (10.18) is solved without computing inverse matrices, in this manner, computational problems such as numerical underflow/overflow are avoided. To see that the previous update formula corresponds to a local-maximum, note that the second derivative of Eq. (10.11) with respect to $\boldsymbol{M}_{im}$ is:

$$\frac{\partial^2 \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})}{\partial \boldsymbol{M}^2_{im}} = -\left(\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) (\overline{\boldsymbol{u}}^t_{iml})^\top \overline{\boldsymbol{u}}^t_{iml} \right) \tag{10.20}$$

Which is a weighed sum of covariance matrices, which, due to the negative sign, results in a negative-semidefinite matrix, and therefore a local-maximum. If $\hat{\mu}^t_{lim} = x^l_m + \boldsymbol{M}^{(s+1)}_{im}(\overline{\boldsymbol{u}}^t_{iml})^\top$, the updating formula for h_{im} is:

$$\frac{\partial \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})}{\partial h_{im}} = \sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \frac{\partial}{\partial h_{im}} \left(\ln \left(\frac{1}{h_{im}} K \left(\frac{x^t_m - \hat{\mu}^t_{lim}}{h_{im}} \right) \right) \right) = 0 \tag{10.21}$$

$$\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \frac{\partial}{\partial h_{im}} \left(-\frac{\left(x^t_m - \hat{\mu}^t_{lim}\right)^2}{2h^2_{im}} - \ln(h_{im}) \right) = 0 \tag{10.22}$$

$$\sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \left(\frac{\left(x^t_m - \hat{\mu}^t_{lim}\right)^2}{h^3_{im}} - \frac{1}{h_{im}} \right) = 0 \tag{10.23}$$

$$\sum_{t=p^*}^{T} \gamma^t(i) \frac{1}{h_{im}} = \sum_{t=p^*}^{T}\sum_{l\neq t}^{T} \psi^t_l(i) \frac{\left(x^t_m - \hat{\mu}^t_{lim}\right)^2}{h^3_{im}} \tag{10.24}$$

$$h_{im}^{(s+1)} = \left(\frac{\sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i)(x_m^t - \hat{\mu}_{lim}^t)^2}{\sum_{t=p^*}^{T} \gamma^t(i)} \right)^{\frac{1}{2}} \tag{10.25}$$

In the Eq. (10.24), we have used that $\gamma^t(i) = \sum_{l \neq t}^{T} \psi_l^t(i)$. To see whether the previous parameter estimate corresponds to a local maximum, the second derivative is computed:

$$\frac{\partial^2 \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})}{\partial h_{im}^2} = -\sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \frac{\psi_l^t(i)}{h_{im}^2} \left(\frac{3\left(x_m^t - \hat{\mu}_{lim}^t\right)^2}{h_{im}^2} - 1 \right). \tag{10.26}$$

In this case, the second derivative is not always negative or non positive. To ensure that the second derivative is negative observe that:

$$-\sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \frac{\psi_l^t(i)}{h_{im}^2} \left(\frac{3\left(x_m^t - \hat{\mu}_{lim}^t\right)^2}{h_{im}^2} - 1 \right) < 0 \tag{10.27}$$

$$\sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \frac{\psi_l^t(i)}{h_{im}^2} < \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i) \left(\frac{3\left(x_m^t - \hat{\mu}_{lim}^t\right)^2}{h_{im}^4} \right) \tag{10.28}$$

$$h_{im}^2 < \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i) \left(3\left(x_m^t - \hat{\mu}_{lim}^t\right)^2 \right) \tag{10.29}$$

$$h_{im} < \left(3 \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i) \left(x_m^t - \hat{\mu}_{lim}^t\right)^2 \right)^{\frac{1}{2}} \tag{10.30}$$

In Eq. (10.30) it is observed that h_{im} corresponds to a local-maximum, if and only if it is lower than the root of the weighted mean of the squares of the data deviations from the kernel centers (scaled by $\sqrt{3}$). It is relevant to mention that the weights are given by the aposterioris $\psi_l^t(i)$. Finally, using the constraint that $\sum_{l \neq t}^{T} \omega_{il} = 1$ and adding a Lagrange multiplier ν_i, the update formula for ω_{il} is:

$$\begin{aligned}
&\frac{\partial \mathcal{Q}(\lambda|\lambda')}{\partial \omega_{il}} = \frac{\partial}{\partial \omega_{il}} \left(\sum_{t=p^*}^{T} \psi_l^t(i) \ln(\omega_{il}) + \nu_i (1 - \sum_{l \neq t}^{T} \omega_{il}) \right) = 0 \\
&\sum_{t=p^*}^{T} \psi_l^t(i) \frac{1}{\omega_{il}} - \nu_i = 0 \Rightarrow \nu_i = \sum_{t=p^*}^{T} \gamma^t(i) \\
&\sum_{t=p^*}^{T} \psi_l^t(i) \frac{1}{\omega_{il}} = \sum_{t=p^*}^{T} \gamma^t(i) \\
&\omega_{il}^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_l^t(i)}{\sum_{t=p^*}^{T} \gamma^t(i)}
\end{aligned} \tag{10.31}$$

To prove that this solution correspond to a local-maximum, observe that the second derivative

is less or equal to zero:

$$\frac{\partial^2 \mathcal{Q}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)})}{\partial \omega_{il}^2} = -\sum_{t=p^*}^{T} \psi_l^t(i) \frac{1}{\omega_{il}^2} \leq 0. \tag{10.32}$$

The previous results can be summarized in the following lemma:

Lemma 10.1. Let $W_l^l = 0$ for $l = 0, ..., L$, and assume that during the (s) iteration of the EM algorithm, $h_{im} < \left(3 \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i) \left(x_m^t - \hat{\mu}_{lim}^t\right)^2\right)^{\frac{1}{2}}$ holds. The previous conditions are necessary and sufficient for the update equations in Eq. (10.19), Eq. (10.25) and Eq. (10.31) to be local-maximum parameters of Eq. (10.11), during an EM iteration.

Since the probabilistic conditions for the hidden variable Q^t are not modified from the traditional HMM, the update formulas for parameters $\mathbf{A}$ and $\boldsymbol{\pi}$ are the same as those found in standard traditional articles such as Rabiner [1990].

10.1.3 Learning the context-specific Bayesian networks

For the SEM algorithm, we are required to optimize the following criterion

$$\begin{aligned} &\mathcal{Q}(\mathcal{B}, \boldsymbol{\lambda}|\mathcal{B}^{(s)}, \boldsymbol{\lambda}^{(s)}) = \\ &\quad E_{P(\boldsymbol{Q}^{p^*:T}, \boldsymbol{W}^{Pp^*:T}|\boldsymbol{X}^{0:T}; \mathcal{B}^{(s)}, \boldsymbol{\lambda}^{(s)})}[\ln f_{\boldsymbol{X}^{p^*:T}, \boldsymbol{Q}^{p^*:T}, \boldsymbol{W}^{p^*:T}}(\boldsymbol{x}^{p^*:T}, \boldsymbol{Q}^{p^*:T}, \boldsymbol{w}^{p^*:T}; \mathcal{B}, \boldsymbol{\lambda})] \\ &\quad - 0.5\#(\mathcal{B}) \ln(T). \end{aligned} \tag{10.33}$$

In this manner complex structures are penalized to prevent overfitting issues and and overly dense Bayesian networks. Due to the linear nature of Eq. (10.11), only the score in Eq. (10.34) is affected by structure modification (for a given state $Q^t = i$ and value for X_m, with the other terms remaining unaffected.

$$\text{score}_{im} = \sum_{t=p^*}^{T} \sum_{l \neq t}^{T} \psi_l^t(i) \ln\left(K\left(\frac{x_m^t - \mu_{lim}^t}{h_{im}}\right)\right) - \frac{1}{2}(\kappa_{im} + p_{im} + T + 1 - p^*) \ln(T). \tag{10.34}$$

For the SEM algorithm, during the search for new structures, the score in Eq. (10.34) is then used with the greedy-forward algorithm described in Section 5.6. However, we emphasize that any other heuristic or meta-heuristic can used for the model search, such as the tabu search in Bueno et al. [2017].

10.1.4 Theoretical computation time complexity

It is well known that the methods based on kernel density estimation can be time demanding. This section therefore provides upper-bounds from the running time of each learning algorithm in terms of big O notation. For these bounds, two scenarios are considered: One where the Bayesian Networks are dense such that no additional arc can be put into the networks,

and other where all the context-specific Bayesian networks are naïve (i.e., they contain zero arcs). We refer to the latter bound as the *light computational upper bound*.

Learning algorithm step	Upper-bound	Light upper-bound
1. Compute $b_i(\boldsymbol{x}^t)$	$O(T^2NMS)$	$O(NT^2M)$
2. Estimate $\gamma^t(i)$, $\psi_l^t(i)$	$O(TN(N^2+T))$	$O(TN(N^2+T))$
3. Update $\mathbf{A}$ and $\boldsymbol{\pi}$	$O(NT)$	$O(NT)$
4. Update $\boldsymbol{\omega}$	$O(NT^2)$	$O(NT^2)$
5. Update $\boldsymbol{M}$	$O(NS^2(T^2+MS))$	$O(0)$
6. Update $\boldsymbol{h}$	$O(NMT^2)$	$O(NMT^2)$

Table 10.1: Computational upper bounds. In the upper-bound column, it is assumed that the context-specific Bayesian networks are dense. In the light upper-bound column, it is assumed that the context-specific Bayesian networks are naïve-Bayesian networks. $S = p^* + M$

Table 10.1 reports the bounds, where for the sake of space, $S := p^* + M$. The bounds are arranged in the same order as the respective steps in the learning algorithm. It is noticeable that the presence of Bayesian networks in the model only affects the computation of emission probabilities and the updates $\boldsymbol{M}$. Nevertheless, the computational effort needed to update $\boldsymbol{M}$ can be high in the case of dense networks, since it is quadratic in the training input length and fourth-power in the number of variables. The fourth-power dependency in the number of variables comes from the solution of linear systems and the loop through all the variables to solve their corresponding system. In this sense, it is desirable to keep the number of dependencies as low as possible; otherwise, the computational time required to train a model can be prohibitive.

The log-likelihood of test data can be evaluated using steps 1 and 2 from Table 10.1. Step 2 implicitly uses the forward-backward algorithm for the computation of $\gamma^t(i)$, which is traditionally used to compute log-likelihoods (specifically, the forward part).

10.1.5 Model initialization

As seen in the previous section, the more complex the context-specific Bayesian networks are, the greater the computational complexity of the learning and inference algorithms. At the start of training it is therefore assumed that all the Bayesian networks are naïve Bayesian networks, i.e., $\kappa_{im} = 0$ and $p_{im} = 0$ for all variables and hidden states. The parameters $\{h_{im}\}_{i=1,m=1}^{N,M}$ are set following the rule of thumb provided by Silverman [1986]: $h_{im} = (4\hat{\sigma}_m^5/(3T))^{\frac{1}{5}}$, $i = 1, ..., N$, where $\hat{\sigma}_m$ is the sample standard deviation of variable X_m. Each parameter $\{\omega_{il}\}_{i=1,l=p^*}^{N,T}$ is initialized using random draws from a uniform distribution on $[0.4, 0.6]$, followed by normalization to satisfy the constraint $\sum_{l=p^*}^{T} \omega_{il} = 1$. The $\boldsymbol{\pi}$ parameter is initially assumed to be a uniform categorical distribution, whereas the parameter $\mathbf{A}$ is set as a matrix whose diagonal is filled with value arbitrary far from 1 (in our case 999), and the remaining values are $1/N$. The matrix is then normalized to be an actual transition matrix. This matrix can clearly be modified as needed in case of left-to-right transition matrices or uniform ones. The initialization described here is used to condition the model to look for

stable patterns.

10.2 Experimental setup

For the experiments, synthetic data and real stochastic data from ambient audio and CNC machines was used to demonstrate the abilities of the proposed model. Our model was compared against a traditional HMM, where all the variables are assumed to be independent Gaussians. Since the proposed model can be seen as a KDE extension of the AR-AsLG-HMM model in Puerto-Santana et al. [2022b], that model was also included in the comparisons. In Puerto-Santana et al. [2022b], AR-AsLG-HMM showed similar or better results when compared with MoG-HMMs, even when synthetic MoG data was considered. Therefore, MoG-HMMs was not included in the experiments.

If structural optimization is not performed, our KDE-AsHMM model can be seen as a multivariate version of the Piccardi and Perez [2007] model when all variables are independent, therefore this model, in the case of assuming independent multivariate data, was also compared. We denote this model KDE-HMM. Finally, ablations of the proposed model were also considered for the synthetic data: KDE-BNHMM is the model where Bayesian networks were built with no AR structural optimization, while KDE-ARHMM is the model where only AR structural optimization was performed. Also, a model called KDE-AsHMM* was compared. This model was provided with the ground truth about AR and non-AR dependency structure, and thus represents performance that might not be attainable in practice when the structure is unknown. The differences between the models are stated in Table 10.2.

Model	Kernel-based	AR	Non-AR	Ground-truth
HMM	-	-	-	-
AR-AsLG-HMM	-	✓	✓	-
KDE-HMM	✓	-	-	-
KDE-AsHMM	✓	✓	✓	-
KDE-ARHMM	✓	✓	-	-
KDE-BNHMM	✓	-	✓	-
KDE-AsHMM*	✓	✓	✓	✓

Table 10.2: Differences in the models used for validation

10.2.1 Synthetic data

10.2.1.1 Data description

For the synthetic data, seven variables were used. It was assumed that the data jumps between three hidden states, and each hidden state having its own context-specific Bayesian network representation as pictured in Figure 10.2. One of the states was assumed to be associated with a naïve Bayes model, hence its graph is not pictured. The names of the variables go from 0 to 6; however, X_5 and X_6 were assumed to be Gaussian noise which turned them independent

of the hidden state variable Q^t, and therefore, they were not related to any variable for any hidden state. We use $m : \mathrm{AR}_r$ to denote an arbitrary variable in the network:, is the r AR order of the variable X_m^t or X_m^{t-r}. It was assumed that each variable X_m had the following

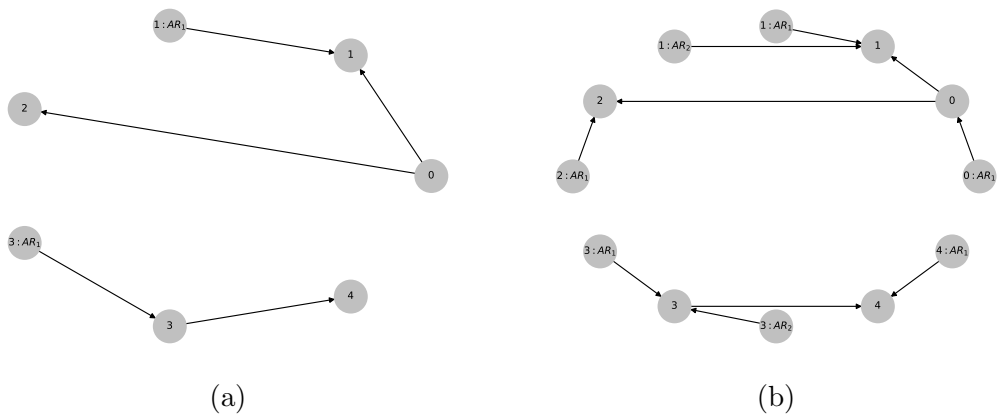


Figure 10.2: The context specific graphs corresponding to the synthetic data. (a) is a simpler version of the Bayesian network pictured in (b)

distribution:

$$X_m^t | Q^t = i \sim \mathcal{N}\left(\sum_{k=1}^{\kappa_{im}} c_{imk}((V_{imk}^t)^2 - e_{im}) + \sum_{r=1}^{p_{im}} d_{imr} X_m^{t-r}, \sigma_{im}^2\right) \tag{10.35}$$

The coefficients $\{\{c_{imk}\}_{k=1}^{\kappa_{im}}, \{d_{imr}\}_{r=1}^{p_{im}}, e_{im}, \sigma_{im}\}_{i=1,m=1}^{N,M}$ are provided in the complementary material. The data was generated from a pre-defined sequence of hidden states which is pictured in Figure 10.3. For the training process, in order to observe how T,the amount of training data, affects model performance, the previous sequences was expanded as needed to obtain datasets of length $T \in \{350, 700, 1050, 1400, 1750, 2100, 2450\}$. For each T, a single time series of that length was sampled and used as training data. Later, one hundred samples with $T_{\text{test}} = 1400$ were generated and used as test dataset. To compare the models, the mean log-likelihood per unit datum and its standard deviation on the testing data were reported.

10.2.1.2 Results

Table 10.3 reports the mean and standard deviation of the log-likelihood per unit datum on the testing data. Note that, when the length of the training data was small ($T = 350$), linear Gaussian models as the AR-AsLG-HMM and HMM could outperform the log-likelihoods reached by the corresponding KDE models such as KDE-AsHMM and KDE-HMM. Nonetheless, when the length of the training data sequence increased to $T = 2450$, it was observed that all the kernel models obtained a better mean log-likelihood per unit data than the linear Gaussian models, with the exception of the KDE-HMM. This shows that the introduction of context-specific Bayesian networks into the kernel models is beneficial to improve how their fitness scales to longer sequences. Regarding the standard deviation of such log-likelihoods

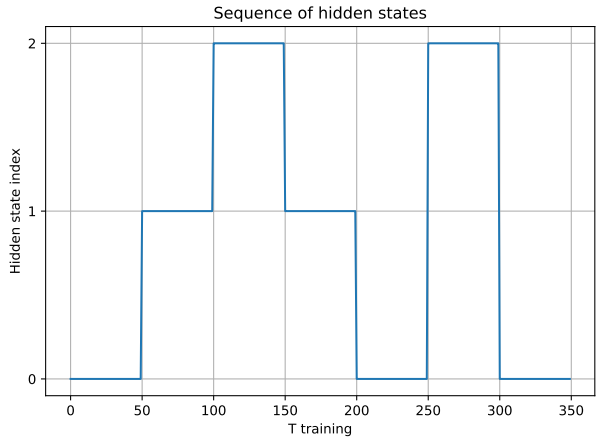


Figure 10.3: Sequence of hidden states used to generate training and testing data. The length of this sequence is expanded proportionally as needed for the training data, maintaining the pattern

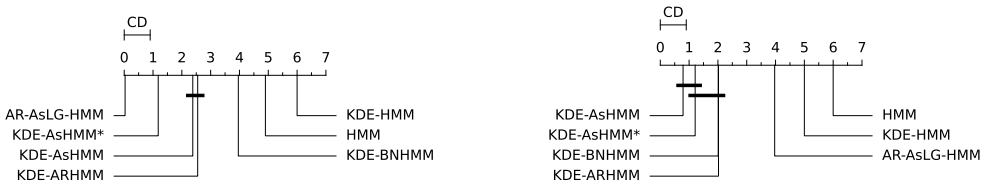
per unit data, it was observed that AR-AsLG-HMM and HMM obtained the lowest values, indicating that their fitness values were more stable than those obtained by kernel-based HMMs.

	$T = 350$			$T = 2450$		
Model	μ	σ	s	μ	σ	s
HMM	-10.52	0.16	**0.24**	-10.36	0.10	**2.07**
AR-AsLG-HMM	**-9.05**	**0.11**	3.64	-8.82	**0.09**	9.98
KDE-HMM	-12.07	0.70	1.81	-9.79	0.11	367.15
KDE-AsHMM	-9.68	0.24	33.60	**-8.11**	0.24	3351.38
KDE-ARHMM	-9.74	0.40	17.69	-8.17	0.25	2229.66
KDE-BNHMM	-9.95	0.47	62.53	-8.17	0.26	5160.31
KDE-AsHMM*	-9.52	0.39	7.46	-8.13	0.31	827.93

Table 10.3: Mean log-likelihood per unit datum and its standard deviation on the testing data. T refers to the size of the length of the training data. Onlyresults for the shortest and longest training sequences. "s" is for seconds.

With respect to the training times, it can be seen that the Gaussian parametric models were faster than their kernel counterparts. However, the simplest KDE-HMM even in the large datasets, was not excessively high-time demanding. Their asymmetric counterparts, KDE-AsHMMs, KDE-BNHMMs and KDE-ARHMMs required up to 60 times more computation time. In the case of KDE-AsHMM*, as the network is already given, the structural optimization was omitted and the training times were up to 4 times greater than the KDE-HMM. Observe that KDE-AsHMM* obtained results close to those obtained by KDE-AsHMM with less computational effort. In this sense, using expert knowledge to build possible dependency graphs is recommended, such that competitive likelihoods can be attained with less computational cost.

To take into account the effects of variance in the likelihoods and to provide a statistical



(a) Nemenyi test $T = 350$ (b) Nemenyi test $T = 2450$

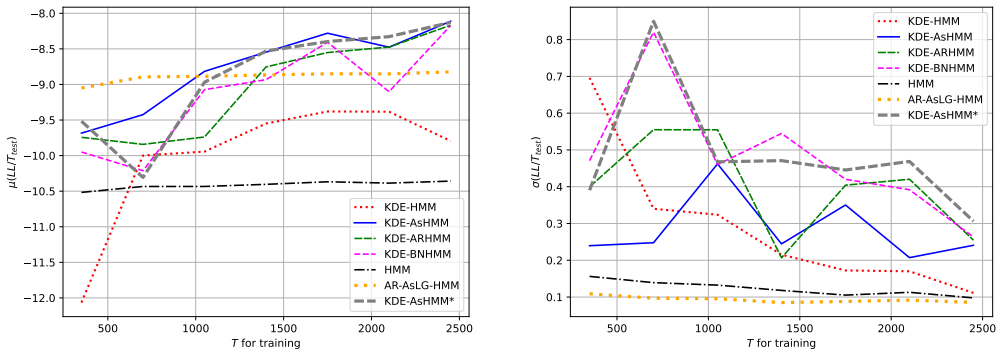
Figure 10.4: Nemenyi ranking test when (a) the training data length is $T = 350$, and (b) the training data length is $T = 2450$. Rankings closer to 0 imply a better fit to testing data

analysis of the model ranking, the Friedman and the post-hoc Nemenyi hypothesis test was used, see Demšar [2006]. The Friedman test checks the hypothesis of global no difference in rankings. If the hypothesis is rejected the post-hoc Nemenyi test is applied. For all pairs of models, the method subsequently tests the hypothesis of no difference in mean ranking in terms of the mean log-likelihood per datum. A critical difference (CD) value is computed to indicate the minimum distance in the rank needed to declare evidence of statistical difference.

The Friedman test when $T = 350$ provided a test statistics of 712.53 which corresponds to a virtual p-value of 0.0. Therefore, the null hypothesis of no difference in ranking between the methods was rejected. When $T = 2450$, the test statistics was 663.63 which again gave a p-value of 0.0. Since null hypothesis was rejected in both cases, we proceeded with the Nemenyi post-hoc tests.

The results of the Nemenyi tests are provided in Figure 10.4. In (a), the test was applied when the training data length was $T = 350$, and it was observed that the top ranked models were AR-AsLG-HMM and KDE-AsHMM*, and the worst were KDE-HMM and HMM. Note that there is no statistical evidence to claim that KDE-AsHMM and KDE-ARHMM were different in the ranking position. Nevertheless, in (b), when $T = 2450$, the best ranked models were KDE-AsHMM and KDE-AsHMM*, and the worst were again HMM and KDE-HMM. However, there was no statistical evidence to claim that KDE-AsHMM and KDE-AsHMM* differed in their ranking. On the other hand, the HMM and KDE-HMM models presented clear evidence of differences in their ranking position.

Figure 10.5 expands on the previous results in mean and standard deviation of log-likelihood per unit datum. In this case, the results when the training data length $T \in \{350, 700, 1050, 1400, 1750, 2100, 2450\}$ are reported in plot (a) and (b), where (a) shows the evolution of the mean log-likelihood per unit datum and (b) pictures its standard deviation. In (a), it was observed that AR-AsLG-HMM was capable of obtaining the best results in terms of likelihood when the training data was small. However, it is also evident from the plot how the performance of the non-parametric kernel models kept improving as more training data became available, unlike the parametric models (HMM and AR-AsLG-HMM). It is worthy to note that, regarding kernel models, which of KDE-AsHMM* and KDE-AsHMM that obtained the best log-likelihood differed for different T values. Regarding (b), Gaussian parametric models, in spite of not getting benefits in likelihood with increases in training data, they did have lower standard deviation, but the variance reductions were not substan-



(a) (b)

Figure 10.5: In (a) the mean log-likelihoods per datum when the size T of training data changes and in (b) the corresponding standard deviation

tial. On the other hand, KDE-HMM greatly reduced variance when T increased, but this did not translate into relevant improvements in terms of log-likelihood, at least, enough to outperform AR-AsLG-HMM. The proposed model KDE-AsHMM and its ablations, showed slightly reduced variance with increasing T, but we do not have statistical evidence to claim that this trend is statistically significant.

To summarize the results on synthetic data, the kernel models in the HMM framework (with the exception of KDE-HMM) all were able to take advantage of the increase in the training dataset size, and the addition of context-specific Bayesian networks improved model performance, but longer training times and greater variance in fitness could be observed as well.

10.2.2 Real data from environmental sound classification

10.2.2.1 Data description

For these experiments, the *Environmental Sound Classification 50*[2] dataset was used. The dataset consists of environmental sounds of fifty different sources as cats, dogs, keyboards, snoring, mouse click, etc. Each sound is measured at a sample rate of 16kHz during 5 seconds. From the raw audio files, 5 mel frequency spectrum coefficients (MFCC), see Davis and Mermelstein [1980], were extracted. The time window had a range of 0.1 seconds or 1600 time instances and the window was sliced every 0.05 seconds or 800 time instances. The dataset was divided into 5 folds, where each fold had eight recordings for each class. Therefore, for the model validation, a 5-fold cross-validation was performed. In this use case, we are concerned with identifying which model, among the ones considered, obtains the best results in terms of accuracy. Therefore, for each fold, for each type model and for each class, a model was learned. The prediction in testing phase was done by selecting the class of the model which maximized the log-likelihood of the data. In the case of the KDE-AsHMM, to

[2]https://www.kaggle.com/datasets/mmoreaux/environmental-sound-classification-50

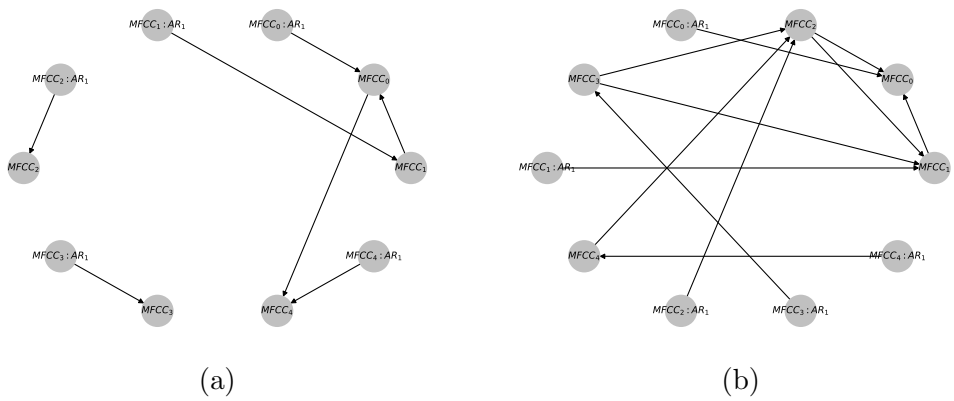
prevent long computational times, the SEM algorithm was iterated only once and p^* was fixed as $p^* = 1$.

10.2.2.2 Results

Model	F1(%)	F2(%)	F3(%)	F4(%)	F5(%)	mean(%)
HMM	22.3	18.3	18.8	26.3	18.5	20.8
AR-AsLG-HMM	28.0	29.3	29.8	30.3	31.8	29.8
KDE-HMM	13.5	15.8	17.0	18.5	13.3	15.6
KDE-AsHMM	**32.3**	**35.3**	**36.3**	**41.8**	**32.3**	**35.6**

Table 10.4: Fold and mean accuracy for the 50 ambient sounds classification problem

In this case only 4 models were compared due to the computational cost: they were HMM, AR-AsLG-HMM, KDE-HMM and KDE-AsHMM. The ablations were omitted, since in the synthetic data was observed that such models can have the equal or worse performance than KDE-AsHMM. Since for each fold a model must be learned, and there are 50 classes, for the classification task, 1000 models had to be learned in total. All models assumed that there are three hidden states within the MFCC training data. A random classification of the testing files, would obtain in mean an accuracy of 2%. Therefore, this can be seen as the lowest tolerable or baseline accuracy for a classifier. In Table 10.4, the results of classification from each model for each fold are provided. As it can be seen, for all the folds, the model with the highest accuracy was the proposed model KDE-AsHMM, followed by AR-AsLG-HMM, HMM and KDE-HMM. The latter had the lowest accuracy, which implies that the addition of kernel corrections and information sharing via Bayesian networks can be helpful to have more accuracy in this classification problem. Finally, in spite that the accuracies were not over 50%, they were at least 5 times higher than the baseline accuracy and at most 20 times higher, which indicates that the models perform fairly better than random prediction.



(a) (b)

Figure 10.6: Bayesian networks obtained from the pig class at fold 1 from KDE-AsHMM. (a) and (b) represent two different hidden states

Recall that the proposed model introduces context-specific Bayesian networks into KDE-

HMMs, which are used to provide kernel corrections, based on the information from AR values and within variables dependencies. In Figure 10.6, two context-specific Bayesian networks from the pig audio KDE-AsHMM are pictured. In (a), AR values were present to explain the MFCC amplitudes and some dependencies as the 4th MFCC depending on the 0th MFCC. In (b), it can be seen that most of the previous relationships hold but further relationships appear, for example, the 2nd MFCC depends on the 4th and 3rd MFCC. These relationships can provide further insights from the learned audio and the sound generation for the class instance, in this case, a pig.

10.2.3 Real data from CNC mill tool

10.2.3.1 Data description

In this case, the *CNC mill tool wear* dataset[3] dataset was used. A series of machining experiments were run on 5.08cm×5.08cm×3.81cm wax blocks in a CNC milling machine in the System-level Manufacturing and Automation Research Testbed (SMART) at the University of Michigan. Machining data was collected from a CNC machine for variations of tool condition, feed rate, and clamping pressure. Each experiment produced a finished wax part with an "S" shape carved into the top face. The data contains samples where the extracted S shaped part had a desired quality and others where such quality was not reached. The quality of a part was determined by eye-quality-controls.

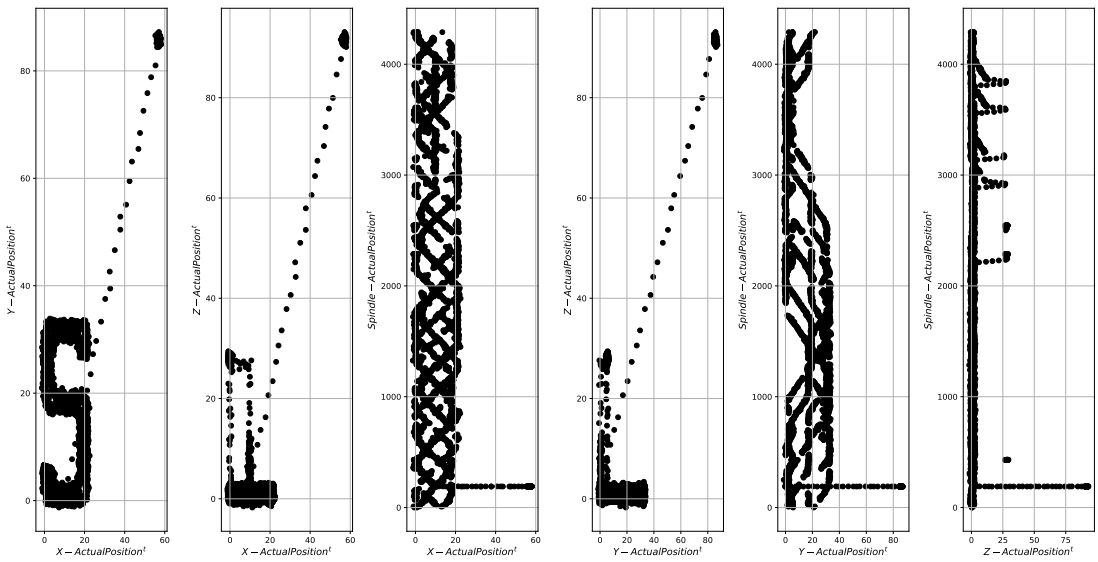


Figure 10.7: Scatter plots for all pairs of selected features. In the X-Y actual position plane, the S-shape extracted piece from the wax block is observed. In the remaining plots, non linear data is observed as in the case of X-spindle actual position plane or Y-spindle plane

In Table 10.5 a brief description of the 18 experiments is provided. Since it is observed that a worn tool can provide an accepted S-piece, this study will focus on determining the fitness of

[3]https://www.kaggle.com/datasets/shasun/tool-wear-detection-in-cnc-mill

the models to describe accepted pieces and non-accepted pieces. The dataset has 44 features related to each axis of the part: its position, velocity, acceleration, current, voltage and power; whereas from the spindle, its position, velocity, acceleration, current, voltage, power and inertia were also recorded. However, during the experiments, it was seen from scatter plots that the variables which showed the most complex data form came from the features related to the position of the spindle and the piece, the remaining variables were constant or more related to Gaussian behavior. Therefore, only 4 variables were used, namely: *actual X-axis position of the part*, *actual Y-axis position of the part*, *actual Z-axis position of the part* and *actual position of the spindle*. Figure 10.7 shows all the scatter plots regarding all pairs of used variables. Note how, in the X-Y actual position plane, the S-shape figure is observed.

Essay	**Tool condition**	**Experiment ended?**	**Accepted?**	**Length**
1	unworn	yes	yes	1055
2	unworn	yes	yes	1668
3	unworn	yes	yes	1521
4	unworn	no	no	532
5	unworn	no	no	462
6	worn	yes	no	1296
7	worn	no	no	565
8	worn	yes	no	605
9	worn	yes	no	740
10	worn	yes	no	1301
11	unworn	yes	yes	2314
12	unworn	yes	yes	2276
13	worn	yes	yes	2233
14	worn	yes	yes	2332
15	worn	yes	yes	1381
16	worn	no	no	602
17	unworn	yes	yes	2150
18	worn	yes	yes	2253

Table 10.5: Description for *CNC mill tool wear* dataset

Observe that the essays 1, 2, 3, 11, 12, 13, 14, 15, 17 and 18 were successful, from these 10 samples, 5 folds were created. Namely, *Fold 1* uses essays 1 and 13, *Fold 2* uses 2 and 14, *Fold 3* uses 3 and 15, *Fold 4* uses 11 and 17, *Fold 5* uses 12 and 18. From the samples in each fold, a model of HMM, AR-AsLG-HMM, KDE-HMM and KDE-AsHMM was trained. The instances of the other four folds were used for testing. In this manner, the fitness of the models with respect to accepted pieces was tested. On the other hand, the essays 4, 5, 6, 7, 8, 9, 10 and 16 obtained non-accepted pieces. All these instances were evaluated for all the folds and models, so that, fitness of non-accepted pieces can be compared. For fitness, the log-likelihood per unit data was used and reported. In this case again $p^* = 1$ and the SEM is also iterate only once. In each dataset, every time instance is labeled with a processing state. From those, only the states regarding the phases: 'Layer 1 Down','Layer 1 Up', 'Layer 2 Up',

'Layer 2 Down', 'Layer 3 Down', 'Layer 3 Up' and'Repositioning'. As there are 7 states, 7 hidden states are used in each model. This information is used to determine the initial values of the $\boldsymbol{\omega}$ parameters. Assume that a matching between hidden states and machining states is created. Then, set $\omega_{il} = 1$ if the machining state of the l instance correspond to the i state, otherwise, set $\omega_{ill} = 1e - 5$. Next, normalize $\boldsymbol{\omega}$ in order to obtain a valid parameter.

10.2.3.2 Results

	Fold 1			**Fold 2**		
Model	*Good*	*Bad*	*Quot.*	*Good*	*Bad*	*Quot.*
HMM	-161.19	-314.15	**1.95**	**-18.22**	-23.2	1.27
AR-AsLG-HMM	-213.01	-182.85	0.86	-361.99	-1544.22	**4.27**
KDE-HMM	-220.58	-234.55	1.06	-85.12	-196.60	2.31
KDE-AsHMM	**-30.75**	-22.29	0.72	-24.14	-47.92	1.99
	Fold 3			**Fold 4**		
Model	*Good*	*Bad*	*Quot.*	*Good*	*Bad*	*Quot.*
HMM	-16.26	-21.17	1.3	-130.77	-255.94	**1.96**
AR-AsLG-HMM	**-8.83**	-18.91	2.14	-142.98	-103.05	0.72
KDE-HMM	-55.37	-128.6	**2.32**	-230.99	-259.58	1.12
KDE-AsHMM	-21.44	-46.46	2.17	**-40.61**	-39.80	0.98
	Fold 5			**Mean**		
Model	*Good*	*Bad*	*Quot.*	*Good*	*Bad*	*Quot.*
HMM	-24.71	-48.84	1.98	-70.23	-132.66	1.69
AR-AsLG-HMM	-121.97	-87.76	0.72	-169.75	-387.36	1.74
KDE-HMM	-81.82	-157.51	**1.93**	-134.78	-195.37	**1.75**
KDE-AsHMM	**-19.83**	-21.05	1.06	**-27.35**	-35.50	1.38

Table 10.6: Log-likelihood per unit data for each model and fold. The columns corresponding to *Good*,(*bad*) refer to the mean fitness of the models regarding accepted (non-accepted) pieces. The columns corresponding to *Quot.*, refer to the quotient *Bad*/*Good*. For the set of results in *Mean*, the mean values across the folds are averaged

In Table 10.6 the obtained results regarding log-likelihood per unit data for each fold are shown. For each fold three columns are displayed, namely: *Good* which refers to the mean fitness obtained from other folds on accepted-pieces time series, *Bad* which refers to the mean fitness obtained when evaluating the non-accepted-pieces time series, and *Quot.* that is the result of dividing *Bad* by *Good*. This last column can be interpreted as a measure of the quality of the model to differentiate between accepted pieces and non accepted pieces. Thus, values above one, implies that the model is able to determine a non-acceptable piece. The column *Mean*, has the mean results for all the columns for all the folds, and therefore its *Quot.* column is not the quotient of its *Bad* and *Good* columns.

From the results, it is observed that our proposed model obtained the best likelihood explaining and modeling processes that lead to accepted pieces in all the folds. Regarding the quotient results, or the ability of the models to differentiate between processes that lead to

accepted parts and unaccepted parts, it is observed that for most of the folds, all the models are capable of differentiating between acceptable and non-acceptable pieces, unless in Fold 1 and Fold 5, where AR-AsLG-HMM obtained results lower than one. And in Fold 1 where KDE-AsHMM obtained a quotient lower than one. In the mean column, it is observed that in overall, the KDE-AsHMM obtained the best results in terms of fitness followed by HMM and KDE-HMM. Regarding the quotients, the highest mean quotient was also obtained by KDE-HMM and followed by AR-AsLG-HMM and HMM. This implies that our model was the best to model the data. However, it was robust enough to understand data from non-accepted pieces, but not enough to provide a quotient lower to one.

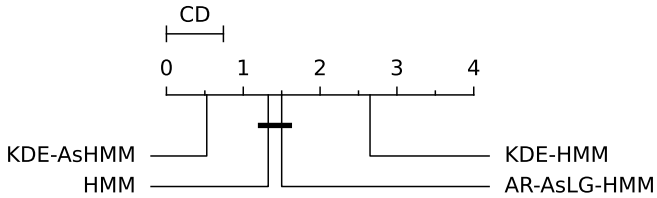


Figure 10.8: Nemenyi hypothesis testing for ranking positions regarding fitness over accepted-pieces signals. The closer to zero, the better the fitness obtained by the model.

To test for statistically significant differences in ranking positions for the four models the Friedman test was applied. The log-likelihoods per unit data obtained by each model across all the folds and accepted-pieces testing signals were ranked in order to perform the test. In this case the test statistic value is 151.29, which leads to a p-value of 0.0. Then, the null-hypothesis can be rejected and the Nemenyi post-hoc test was applied. The results of such test are pictured in Figure 10.8, where it is observed that KDE-AsHMM obtained in mean the best ranking, followed by HMM, AR-AsLG-HMM and KDE-HMM. From the tests, the only models that were not found be statistically different in their rankings were HMM and AR-AsLG-HMM. Therefore, with this test, it can be claimed that the proposed model KDE-AsHMM was the best in terms of fitness.

As summary, the KDE-AsHMM was the best to explain processes that lead to accepted pieces, followed by HMM. Additionally, KDE-AsHMM was also useful to detect or determine if a piece should not be accepted, since in mean a non-accepted piece would increase the fitness the largest when compared with the fitness obtained for an accepted piece.

Finally, recall that our proposed model is capable of providing Bayesian networks that can be used to understand and generate further data insights. In this case, the obtained Bayesian networks for fold 3 are represented in Figure 10.9. In spite that there are 7 hidden states, for the sake of space, only 2 of them are pictured. In (a) we find a Bayesian networks where all the axes rely on their AR values, whereas the spindle axis is only related to the Z axis. Also, the Z, X and Y axes are related. Meanwhile in (b), it is observed that in this case the Spindle, X and Z axes depend on AR values. The spindle axis is not statistically related to any other axis position. Nonetheless, the Z axis is statistically related to the X and Y axes. This implies that depending on the hidden state some axes may be independent

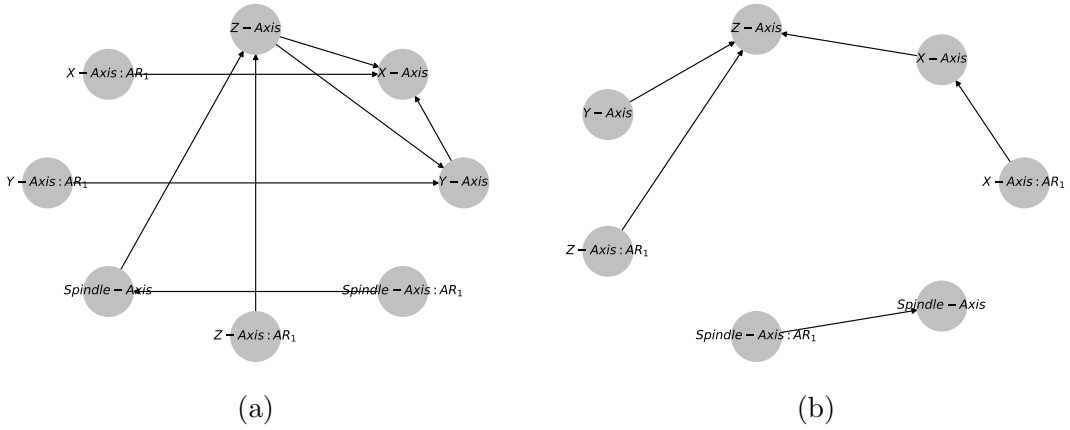


Figure 10.9: Bayesian networks obtained from the *CNC mill tool wear* dattaset

of others and AR values can be relevant to explain the current behavior. It is worthy to note that (a) appeared in 3 out of the 7 hidden states. Therefore, it can be said that this behavior is common and expected for different machining processes.

10.3 Conclusions

In this chapter a new kind of asymmetric HMMs has been proposed. The model introduces kernel corrections via conditional dependencies represented using context-specific Bayesian networks. Its learning procedure is based on the EM and SEM algorithms. Additionally, theoretic computational bounds for the learning and inference algorithms in term of big O notation are given, both in the case of dense networks and in the case of naïve Bayes.

Experiments with synthetic and real data and showed how the proposed models got benefits from using more data in the training phase. It was observed that Gaussian linear Gaussian models can outperform traditional kernel based models in the synthetic data; nonetheless, the proposed model, thanks to the flexibility provided by the kernel corrections, was capable of overpassing Gaussian linear models in terms of log-likelihood per unit data. It was also noted that the computational time can increase drastically during the search of structures for the model, using the kernel correction mechanism. As a consequence, it is advised to provide a set of context-specific Bayesian networks for the models to reduce the computational cost when possible.

For the real life data applications, data from ambient sounds for classification was used to see the performance of the model in terms of accuracy. The baseline accuracy was 2% (50 classes) and it was observed that the proposed model could achieve accuracies up to 45%, which outperformed other tested HMMs. It was also observed that for some cases, Gaussian linear models could perform better than a kernel-based model. In this sense, it was demonstrated again the relevancy of enabling information sharing among the features for a model, (in this case, the information is shared using context-specific Bayesian networks). Finally, data from a CNC machine was studied, where a spindle extracted a S-shape piece

from a wax block. The proposed model obtained the best results in terms of log-likelihood per unit data and was also useful to detect non-acceptable pieces. Also, the learned Bayesian networks could provide further data insights in the form of axes statistical dependencies during machining processes.

In other works such as Atienza et al. [2022], it was observed that in the Bayesian network structure, the nodes were allowed to change from Gaussian linear models to KDE models. In this sense, it would be relevant to check how these transformations can fit in our proposal and how it can be beneficial to reduce the computational time burdens. Additionally, we believe that the current implementation can be improved in order to minimize training times. Also, in traditional KDE, measures as the mean integrated square error (MISE) (see Section 2.1.4) are used to determine the estimation quality (in our case, the likelihood serves as estimation quality). Therefore, a more theoretic research must be done to determine if there is coherence between the MISE score and the EM algorithm.

Part III

CONCLUSIONS

Chapter 11

Conclusions

11.1 Answer to the main thesis question

Recall the thesis question: ***Is it possible to generate, create or define an adaptive model or methodology based on HMMs capable to work online to estimate ball bearings health and RUL?.*** From the state of the art in Section 3.3, it was cleared that such model was already developed, e.g., in Ocak et al. [2007] or Yu [2017]. Nonetheless, it was observed that such models lacked explainability in their proposed health indexes and were sensitive to outliers and small changes in data distribution, which could be related to false alarms, as observed in Chapter 7. Hence, a new methodology based on asymmetric HMMs was proposed. Asymmetric HMMs to work with continuous variables was introduced in Chapters 4 and Chapter 5. They were adapted to work in data-stream environments where no run to failure data was assumed. By the results obtained in Chapter 7, the proposed methodology could provide in a continuous way a more informative health index than others in the state-of-the-art, and it was also capable of providing a remaining useful life estimation. Nonetheless, such methodology lacked a dynamic FSS procedure. As consequence, the model in Chapter 6 was proposed, where the As-HMMs were endowed with the capacity of performing an embedded FSS. Such FSS was done in an offline manner. As a consequence, in Chapter 8 and Chapter 9, As-HMMs with embedded FSS were adapted to work in data streams. Finally, in Chapter 10, the ideas proposes in Chapter 5 were mixed with KDE models to overcome the restriction of Gaussian data. Regarding the functional requirements in Chapter 1, the following can be said:

A. The model must be fast in its learning and inference phases

The first version of the main model used for the solution of the question was introduced in Chapter 4 and improved in Chapter 5. In the latter model, bounds for the learning and inference algorithms were obtained and execution times for different data lengths were provided. It was observed that the times were fair with clear improvements in terms of log-likelihood and the BIC score. Nonetheless, such bounds and time results had to be recomputed in Chapter 7, since the number of hidden states changed in the data-stream. In such chapter, a deeper view of the algorithm and its implementation in terms of time and

memory usage was done. From this analysis, it was observed the weak and strong points of the model implementations. Additionally, recommendations and warnings were declared to ensure that the model can work in data streams.

B. The model or methodology must be able to detect novel concepts in data stream and adapt to them

The novelty detection techniques described in Section 2.4 were applied to work with the proposed models. Chapters 7, 8 and 9, describe how HMMs can be used to detect novel concepts and adapt to them. It was observed that the novelty detection strategies could detect changes in distribution or feature saliencies. Also, the methodologies were robust enough to prevent jumps due to outliers. Nevertheless, this caused lags in the model update, i.e., the model parameters were updated several time instants after the true change had already happened. However, the models were more informative and useful when compared to others in the state of the art.

C. The model or methodology must be able to estimate the industrial asset health and remaining useful life

D. The model or methodology must work in unsupervised environments, since not enough failure labeled data is available

In the next paragraphs, literal **C.** and **D.** are answered. The methodology proposed in Chapter 7 assumes that no previous RTF data is available. Hence, unsupervised data was processed in a data stream fashion. In spite of that, the model was capable of estimating the ball bearing health and its RUL. Regarding the health index, it was useful for giving an approximation of the ball bearing state and it was interpretable, i.e., it measures the amount of orders of magnitude of difference between the current state and an initial healthy state. Also, due to the Page test and the Chernoff bounds, the health index was robust to outliers. Regarding the RUL estimation, it was seen that even when no previous failure data was available, the model was capable of performing fair predictions under some mechanical and sensor conditions. If any of those assumptions was violated, the RUL predictions were poor and non informative. In general, the methodology showed good results in the health index estimation when compared to other state of the art methodologies, but improvements are required to provide more accurate RUL estimations.

E. The model or methodology must determine the most relevant features in an online manner

The methodology used in Chapters 5 and 7 provided a solution to the thesis question. Nevertheless, the solution lacks the FSS phase. To address such issue, the Chapters 6, 8 and 9, introduced models and methodologies to perform an embedded FSS, or let the models themselves determine the most relevant features in an online manner in data stream environments. Chapter 6 enabled the model from Chapter 5 to do an embedded FSS by adding hidden feature saliency variables. Chapter 8 proposed a methodology to use a simpler version of the model from Chapter 6 to perform a FSS in data stream environments. In this case, the set of relevant features changed over time. Chapter 9 introduced an extension of the model proposed in Chapter 6 as in Chapter 7, where the set of relevant variables changed depending

on the hidden state. This allows the model remember the set of previous relevant features after the detection of a novel concept in the data stream. In the experiments, synthetic and real data from ball bearings was used. Irrelevant or noisy variables were detected, and variables which did not provide information were ignored. In spite of the results, the models could not be embedded, during the writing of this thesis, into edge devices for evaluation and testing.

F. The model or methodology must be interpretable

All the proposed models in this thesis aimed to use probabilistic and statistics foundations. Every parameter can be interpreted, and decisions can be justified by means of log-likelihood or BIC. For instance, in the last Chapter 10, context-specific Bayesian networks were embedded into KDE models. In that case, the parameters given by the Bayesian networks were understood as kernel corrections and had an influence in the density mode locations for data recognition and generation. This kind of interpretations can be done for every parameter of all the proposed models, and therefore, any decision made by the models can be inferred from the parameters.

11.2 List of publications

During the development of this thesis, the following publications and submissions were produced:

JCR articles:

- C. Puerto-Santana, P. Larrañaga and C. Bielza, "Autoregressive Asymmetric Linear Gaussian Hidden Markov Models," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 9, pp. 4642-4658, 1 Sept. 2022,.
- C. Puerto-Santana et al., "Asymmetric HMMs for Online Ball-Bearing Health Assessments," in IEEE Internet of Things Journal, vol. 9, no. 20, pp. 20160-20177, 15 Oct.15, 2022,
- C. Puerto-Santana, P. Larrañaga and C. Bielza, "Feature Saliencies in Asymmetric Hidden Markov Models," in IEEE Transactions on Neural Networks and Learning Systems, 2022,
- C. Puerto-Santana, P. Larrañaga and C. Bielza, "Feature Subset Selection in Data-Stream Environments Using Asymmetric Hidden Markov Models and Novelty Detection," in Neurocomputing, Submitted
- C. Puerto-Santana, P. Larrañaga, C. Bielza and G.E Henter, "Context-specific kernel-based hidden Markov model for time series analysis," in ArXiv, Submitted

Proceedings

- Puerto-Santana, C., Bielza, C., Larrañaga, P. (2018). Asymmetric Hidden Markov Models with Continuous Variables. In: , et al. Advances in Artificial Intelligence. CAEPIA 2018. Lecture Notes in Computer Science, vol 11160. Springer, Cham.

- Puerto-Santana, C., Larrañaga, P., Diaz-Rozo, J., Bielza, C. (2022). An Online Feature Selection Methodology for Ball-Bearing Harmonic Frequencies Based on HMMs. In 16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021). SOCO 2021. Advances in Intelligent Systems and Computing, vol 1401. Springer, Cham.

11.3 Future research lines

The current thesis has provided tools and models to enhance the problem of processing and analyzing stochastic dynamical data, with a paramount focus on industrial assets such as ball bearings. Nevertheless, even if determining the health and RUL of the ball bearing issues were handled, several further extensions can be made to the current models in order to be applicable in other industrial problems. For instance, in some industrial applications, the data is registered in uneven time-steps and therefore, the Markovian property of the proposed models may not hold. In such cases, continuous time Markov models can be used to model such data in a more proper way. Also, the proposed KDE-AsHMM was only proposed for offline analysis. The corresponding data stream version of KDE-AsHMM must be generated and adapted to work properly in data-streams, taking into consideration computational issues and time consumption restrictions.

The prediction of RUL might be improved in many directions. One is related to memory allocation. The current model requires the whole history of health indexes to make predictions and a strategy should be found to reduce or collapse data instances ensuring minimal information loss. Other direction concerns model prediction design, since a more robust regression method for prediction is required. The current regression is sensitive to health recoveries or sensor problems.

Finally, in spite that kernel density models were explored to model non-Gaussian data, newer models such a normalizing flows (**?**) and denoising diffusion models (Ho et al. [2020]) are gaining attention due to their flexibility and high accuracy to model arbitrary multidimensional distributions. Then, it would be interesting to explore normalizing flows or denoising diffusion models with asymmetric HMMs in an explainable manner towards solutions for industrial applications.

Bibliography

O. Abdeljaber, S. Sassi, O. Avci, S. Kiranyaz, A. A. Ibrahim, and M. Gabbouj. Fault detection and severity identification of ball bearings by online condition monitoring. *IEEE Transactions on Industrial Electronics*, 66(10):8136–8147, 2019.

S. Adams and P. A. Beling. Feature selection for hidden Markov models with discrete features. In *Intelligent Systems and Applications*, pages 67–82. Springer International Publishing, 2020.

S. Adams, P. Beling, and C. R. Feature selection for hidden Markov models and hidden semi-Markov models. *IEEE Access*, 4:1642–1657, 2016.

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

C. Alexander and M. Sadiku. *Fundamentals of Electric Circuits*. McGraw-Hill, Inc., 3 edition, 2007.

F. Alifers, I. Tsamardinos, and A. Statnikov. HITON: A novel Markov blanket algorithm for optimal variable selection. In *AMIA, Annual Symposium Proceedings*, volume 2003, pages 21–25, 2003.

N. Almusallam, Z. Tari, J. Chan, A. Fahad, A. Alabdulatif, and M. Al-Naeem. Towards an unsupervised feature selection method for effective dynamic features. *IEEE Access*, 9: 77149–77163, 2021.

J. Antoni. Fast computation of the kurtogram for the detection of transient faults. *Mechanical Systems and Signal Processing*, 21(1):108–124, 2007.

A. Asahara, K. Maruyama, and R. Shibasaki. A mixed autoregressive hidden Markov chain model applied to people's movements. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 414–417. ACM, 2012.

D. Atienza, C. Bielza, and P. Larrañaga. Semiparametric Bayesian networks. *Information Sciences*, 584:564–582, 2022.

L. M. Barclay, R. A. Collazo, J. Q. Smith, P. A. Thwaites, and A. E. Nicholson. The dynamic chain event graph. *Electronic Journal of Statistics*, 9(2):2130–2169, 2015.

J. P. Barddal, F. Enembreck, H. M. Gomes, A. Bifet, and B. Pfahringer. Merit-guided dynamic feature selection filter for data streams. *Expert Systems with Applications*, 116: 227–242, 2019.

E. Bechhoefer. Envelope bearing analysis: Theory and practice. *IEEE Aerospace Conference Proceedings*, 2005:3658–3666, 2005.

J. A. Bilmes. Buried Markov models: A graphical-modeling approach to automatic speech recognition. *Computer Speech and Language*, 17(2):213–231, 2003.

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence*, pages 115—-123. Morgan Kaufmann, 1996.

G. Box and G. Jenkins. *Time Series Analysis: Forecasting and Control.* Wiley, 1976.

J. D. Bryan and S. E. Levinson. Autoregressive hidden Markov model and the speech signal. *Procedia Computer Science*, 61:328–333, 2015.

R. G. Budynas, J. K. Nisbett, and J. E. Shigley. *Shigley's Mechanical Engineering Design.* McGraw-Hill, 2011.

M. L. Bueno, A. Hommersom, P. J. Lucas, and A. Linard. Asymmetric hidden Markov models. *International Journal of Approximate Reasoning*, 88:169–191, 2017.

D. L. Cartella F., Lemeire J. and S. H. Hidden semi-Markov models for predictive maintenance. *Mathematical Problems in Engineering*, 2015, 2015.

X. Chang, F. Nie, S. Wang, Y. Yang, X. Zhou, and C. Zhang. Compound rank-k projections for bilinear analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 27(7): 1502–1513, 2016.

C. Chen, N. Lu, B. Jiang, and C. Wang. A risk-averse remaining useful life estimation for predictive maintenance. *IEEE/CAA Journal of Automatica Sinica*, 8(2):412–422, 2021.

Y. Chen, G. Peng, Z. Zhu, and S. Li. A novel deep learning method based on attention mechanism for bearing remaining useful life prediction. *Applied Soft Computing*, 86:105919, 2020.

J. Cheng. A transitional Markov switching autoregressive model. *Communications in Statistics - Theory and Methods*, 45(10):2785–2800, 2016.

H. Chernoff. A measure of asymptotic efficiency for Tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952.

S. Dang, S. Chaudhury, B. Lall, and P. K. Roy. Autoregressive hidden Markov model with missing data for modelling functional MR imaging data. In *Proceedings of the 10th Indian*

Conference on Computer Vision, Graphics and Image Processing, pages 93:1–93:8. ACM, 2016.

M. Dash and Y.-S. Ong. Relief-c: Efficient feature selection for clustering over noisy data. In *2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, pages 869–872, 2011.

S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

J. G. De Gooijer, G. E. Henter, and A. Yuan. Kernel-based hidden Markov conditional densities. *Computational Statistics & Data Analysis*, 169:107431, 2022.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006.

J. Diaz-Rozo, C. Bielza, and P. Larrañaga. Machine-tool condition monitoring with Gaussian mixture models-based dynamic probabilistic clustering. *Engineering Applications of Artificial Intelligence*, 89:103434, 2020.

V. H. Do, X. Xiao, E. Chng, and H. Li. Kernel density-based acoustic model with cross-lingual bottleneck features for resource limited lvcsr. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pages 6–10, 2014.

V. A. Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

C. Fahy and S. Yang. Finding and tracking multi-density clusters in online dynamic data streams. *IEEE Transactions on Big Data*, pages 1–1, 2019a.

C. Fahy and S. Yang. Dynamic feature selection for clustering high dimensional data streams. *IEEE Access*, 7:127128–127140, 2019b.

W. Fan and N. Bouguila. An online approach for learning non-Gaussian mixture models with localized feature selection. In *2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, pages 1–6, 2013.

M. M. M. Farag, T. Elghazaly, and H. A. Hefny. Face recognition system using HMM-PSO for feature selection. In *2016 12th International Computer Engineering Conference (ICENCO)*, pages 105–110, 2016.

G. D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

F. Freitas, S. Peres, C. Lima, and F. Barbosa. Grammatical facial expressions recognition with machine learning. In *Proceedings of the 27th Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pages 180–185, 2014.

N. Friedman. The Bayesian structural EM algorithm. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 129–138. Morgan Kaufmann, 1998.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

J. a. Gama, I. Žliobaitundefined, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 2014.

N. Gebraeel, A. Elwany, and J. Pan. Residual life predictions in the absence of prior degradation knowledge. *IEEE Transactions on Reliability*, 58(1):106–117, 2009.

D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82(1):45–74, 1996.

F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986.

F. Glover and M. Laguna. *Tabu Search*. Springer US, 1997.

B. P. Graney and K. Starry. Rolling element bearing analysis. *Materials Evaluation*, 70: 78–85, 2011.

L. Guerra, C. Bielza, and P. Larrañaga. Semi-supervised projected model-based clustering. *Data Mining and Knowledge Discovery*, 28:882–917, 2014.

M. A. Hall et al. Correlation-based feature selection for machine learning. 1999.

J. D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica*, 57(2):357–384, 1989.

J. D. Hamilton. Analysis of time series subject to changes in regime. *Journal of Econometrics*, 45(1):39–70, 1990.

D. Heckerman. Probabilistic similarity networks. *Networks*, 20(5):607–636, 1990.

G. E. Henter, A. Leijon, and W. B. Kleijn. Kernel density estimation-based Markov models with hidden state, 2018.

D. V. Hinkley. Inference about the change-point from cumulative sum tests. *Biometrika*, 58 (3):509–523, 1971.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

H. Hoeltgebaum, N. Adams, and C. Fernandes. Estimation, forecasting, and anomaly detection for nonstationary streams using adaptive estimation. *IEEE Transactions on Cybernetics*, pages 1–12, 2021.

R. Hofmann and V. Tresp. Discovering structure in continuous variables using Bayesian networks. *Advances in Neural Information Processing Systems*, 8, 1995.

R. V. Hogg and J. W. McKean. *Introduction to Mathematical Statistics.* Pearson Prentice Hall, 2005.

S. C. H. Hoi, J. Wang, P. Zhao, and R. Jin. Online feature selection for mining big data. In *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, pages 93—-100. Association for Computing Machinery, 2012.

H. Huang, S. Yoo, and S. P. Kasiviswanathan. Unsupervised feature selection on data streams. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, page 1031–1040. Association for Computing Machinery, 2015.

F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

J. C. Jáuregui-Correa and A. Lozano-Guzman. *Mechanical Vibrations and Condition Monitoring.* Academic Press, 2020.

R. Jiao, K. Peng, and J. Dong. Remaining useful life prediction for a roller in a hot strip mill based on deep recurrent neural networks. *IEEE/CAA Journal of Automatica Sinica*, 8(7): 1345–1354, 2021.

I. Jolliffe. Generalizations and adaptations of principal component analysis. In *Principal Component Analysis*, pages 223–234. Springer, 1986.

M. C. Jones, J. S. Marron, and S. J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433):401–407, 1996.

B.-H. Juang and L. Rabiner. Mixture autoregressive hidden Markov models for speech signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(6):1404–1413, 1985.

Y. Jung and J. Park. Scalable inference for hybrid Bayesian hidden Markov model using Gaussian process emission. *Journal of Computational and Graphical Statistics*, 0(0):1–18, 2022.

A. B. Kahn. Topological sorting of large networks. *Communications ACM*, 5(11):558–562, 1962.

P. Kenny, M. Lennig, and P. Mermelstein. A linear predictive HMM for vector-valued observations with applications to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(2):220–225, 1990.

M. A. Khan and K. A. Abuhasel. An evolutionary multi-hidden Markov model for intelligent threat sensing in industrial internet of things. *The Journal of Supercomputing*, 77:6236–6250, 2021.

K. Kira, L. A. Rendell, et al. The feature selection problem: Traditional methods and a new algorithm. In *AAAI-92 Proceedings*, volume 2, pages 129–134, 1992.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

S. Kirshner, S. Padhraic, and R. Andrew. Conditional Chow-Liu tree structures for modeling discrete-valued vector time series. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 317–324. AUAI Press, 2004.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

F. Kong, J. Li, B. Jiang, H. Wang, and H. Song. Integrated generative model for industrial anomaly detection via bi-directional LSTM and attention mechanism. *IEEE Transactions on Industrial Informatics*, pages 1–10, 2021.

A. Kumar, R. B. Chinnam, and F. Tseng. An HMM and polynomial regression based approach for remaining useful life and health state estimation of cutting tools. *Computers and Industrial Engineering*, 128:1008–1014, 2019.

P. Larrañaga, D. Atienza, J. Diaz-Rozo, A. Ogbechie, C. Puerto-Santana, and C. Bielza. *Industrial Applications of Machine Learning*. CRC Press, 2018.

M. Law, M. Figueiredo, and A. Jain. Simultaneous feature selection and clustering using mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9): 1154–1166, 2004.

S. Lee, L. Ji, and J. Jun. Online degradation assessment and adaptive fault detection using modified hidden Markov model. *IEEE Transactions on Reliability*, 132(2):183–198, 2010.

H. Li, G. Hu, J. Li, and M. Zhou. Intelligent fault diagnosis for large-scale rotating machines using binarized deep neural networks and random forests. *IEEE Transactions on Automation Science and Engineering*, pages 1–11, 2021.

N. Li, Y. Lei, J. Lin, and S. X. Ding. An improved exponential model for predicting remaining useful life of rolling element bearings. *IEEE Transactions on Industrial Electronics*, 62(12): 7762–7773, 2015.

W. Li and T. Liu. Time varying and condition adaptive hidden Markov model for tool wear state estimation and remaining useful life prediction in micro-milling. *Mechanical Systems and Signal Processing*, 131:689–702, 2019.

Y. Li, M. Dong, and J. Hua. Simultaneous localized feature selection and model detection for Gaussian mixtures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):953–960, 2009.

J. Lin, Z. Lin, G. Liao, and H. Yin. A novel product remaining useful life prediction approach considering fault effects. *IEEE/CAA Journal of Automatica Sinica*, 8(11):1762–1773, 2021.

D. Liu, H. Zhen, D. Kong, X. Chen, L. Zhang, M. Yuan, and H. Wang. Sensors anomaly detection of industrial internet of things based on isolated forest algorithm and data compression. *Scientific Programming*, 2021:1–9, 2021.

A. Luati and M. Novelli. Explicit-duration hidden Markov models for quantum state estimation. *Computational Statistics & Data Analysis*, 158:107183, 2021.

F. Lv and R. Nevatia. Recognition and segmentation of 3-D human action using HMM and multi-class Adaboost. In *Computer Vision – ECCV 2006*, pages 359–372. Springer Berlin Heidelberg, 2006.

Y. Lv, Y. Lin, X. Chen, D. Wang, and C. Wang. Online streaming feature selection based on feature interaction. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 49–57, 2020.

R. Ma, Y. Wang, and L. Cheng. Feature selection on data stream via multi-cluster structure preservation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, page 1065–1074. Association for Computing Machinery, 2020.

N. Malesevic, D. Markovic, G. Kanitz, M. Controzzi, C. Ciprianiand, and C. Antfolk. Vector autoregressive hierarchical hidden Markov models for extracting finger movements using multichannel surface EMG signals. *Complexity*, 2018:0–12, 2018.

M. Markou and S. Singh. Novelty detection: A review—Part 1: Statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

M. M. Masud, Q. Chen, J. Gao, L. Khan, J. Han, and B. Thuraisingham. Classification and novel class detection of data streams in a dynamic feature space. In *Machine Learning and Knowledge Discovery in Databases*, pages 337–352. Springer Berlin Heidelberg, 2010.

N. Michael, D. Metaxas, and C. Neidle. Spatial and temporal pyramids for grammatical expression recognition of American sign language. In *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 75—-82. Association for Computing Machinery, 2009.

V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, pages 2204—2212. MIT Press, 2014.

R. K. Mobley. *An Introduction to Predictive Maintenance*. Butterworth-Heinemann, 2002.

R. Mohammadi-Ghazi, R. E. Welsch, and O. Büyüköztürk. Kernel dependence analysis and graph structure morphing for novelty detection with high-dimensional small size data set. *Mechanical Systems and Signal Processing*, 143:106775, 2020.

M. Momenzadeh, M. Sehhati, and H. Rabbani. A novel feature selection method for microarray data classification based on hidden Markov model. *Journal of Biomedical Informatics*, 95:103213, 2019.

K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

E. Nakamura, P. Cuvillier, A. Cont, N. Ono, and S. Sagayama. Autoregressive hidden semi-Markov model of symbolic music performance for score following. In *16th International Society for Music Information Retrieval Conference* . Archive ouverte HAL, 2015.

P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Morello, N. Zerhouni, and C. Varnier. Pronostia: An experimental platform for bearings accelerated life test. *IEEE International Conference on Prognostics and Health Management*, 2012.

H.-L. Nguyen, Y.-K. Woon, W.-K. Ng, and L. Wan. Heterogeneous ensemble for feature drifts in data streams. In *Advances in Knowledge Discovery and Data Mining*, pages 1–12. Springer Berlin Heidelberg, 2012.

T. D. Nguyen and S. Ranganath. Facial expressions in American sign language: Tracking and recognition. *Pattern Recognition*, 45(5):1877–1891, 2012.

T. M. Nguyen, Q. M. J. Wu, and H. Zhang. Asymmetric mixture model with simultaneous feature selection and model detection. *IEEE Transactions on Neural Networks and Learning Systems*, 26(2):400–408, 2015.

H. Nyman, J. Pensar, T. Koski, and J. Corander. Stratified graphical models - Context-specific independence in graphical models. *Bayesian Analysis*, 9(4):883–908, 2014.

H. Ocak, K. A. Loparo, and F. M. Discenzo. Online tracking of bearing wear using wavelet packet decomposition and probabilistic modeling: A method for bearing prognostics. *Journal of Sound and Vibration*, 302(4):951–961, 2007.

G. Oliveira, L. L. Minku, and A. L. I. Oliveira. Tackling virtual and real concept drifts: An adaptive Gaussian mixture model approach. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021.

J. Omura. On the Viterbi decoding algorithm. *IEEE Transactions on Information Theory*, 15(1):177–179, 1969.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065 – 1076, 1962.

A. Pérez, P. Larrañaga, and I. Inza. Bayesian classifiers based on kernel density estimation: Flexible classifiers. *International Journal of Approximate Reasoning*, 50(2):341–362, 2009.

M. Piccardi and O. Perez. Hidden Markov models with kernel density estimation of emission probabilities and their use in activity recognition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

A. Poritz. Linear predictive hidden Markov models and the speech signal. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 1291–1294, 1982.

C. Puerto-Santana, C. Bielza, and P. Larrañaga. Asymmetric hidden Markov models with continuous variables. In *Procedings of the conference of the Spanish Association for Artificial Intelligence*, pages 98–107. Springer, 2018.

C. Puerto-Santana, C. Bielza, J. Diaz-Rozo, G. Ramirez-Gargallo, F. Mantovani, G. Virumbrales, J. Labarta, and P. Larrañaga. Asymmetric HMMs for online ball-bearing health assessments. *IEEE Internet of Things Journal*, pages 1–1, 2022a.

C. Puerto-Santana, P. Larrañaga, and C. Bielza. Autoregressive asymmetric linear Gaussian hidden Markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4642–4658, 2022b.

C. Puerto-Santana, P. Larrañaga, and C. Bielza. Feature saliencies in asymmetric hidden Markov models. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–0, 2022c.

C. Puerto-Santana, P. Larrañaga, J. Diaz-Rozo, and C. Bielza. An online feature selection methodology for ball-bearing harmonic frequencies based on hmms. In *16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021)*, pages 546–555. Springer International Publishing, 2022d.

Y. Qiao and W. Xi. A kernel density estimation model for moving object detection. In *Advanced Multimedia and Ubiquitous Engineering*, pages 386–392. Springer, 2017.

H. Qiu, J. Lee, J. Lin, and G. Yu. Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. *Journal of Sound and Vibration*, 289(4):1066–1090, 2006.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in Speech Recognition*, pages 267–296. Morgan Kaufmann, 1990.

M. Rajarshi. Bootstrap in Markov-sequences based on estimates of transition density. *Annals of the Institute of Statistical Mathematics*, 42(2):253–268, 1990.

C. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):24–34, 1987.

M. Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27(3):832 – 837, 1956.

W. Rudin. *Principles of Mathematical Analysis.* McGraw-Hill Book Co., third edition, 1976.

Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

M. Sahami. Learning limited dependence Bayesian classifiers. In *KDD-96 Proceedings*, volume 96, pages 335–338, 1996.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

M. Seifert, K. Abou-El-Ardat, B. Friedrich, B. Klink, and A. Deutsch. Autoregressive higher-order hidden Markov models: Exploiting local chromosomal dependencies in the analysis of tumor expression profiles. *PLoS One*, 9(6):1–16, 2014.

R. Setiono and H. Liu. Neural-network feature selector. *IEEE Transactions on Neural Networks*, 8(3):654–662, 1997.

R. D. Shachter. Probabilistic inference and influence diagrams. *Operations Research*, 36(4): 589–604, 1988.

W. Shao, L. He, C.-T. Lu, X. Wei, and P. S. Yu. Online unsupervised multi-view feature selection. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1203–1208, 2016.

B. Silverman. *Densiy Estimation for statistics and data analysis.* Routledge, 1986.

R. K. Singleton, E. G. Strangas, and S. Aviyente. Extended Kalman filtering for remaining-useful-life estimation of bearings. *IEEE Transactions on Industrial Electronics*, 62(3): 1781–1790, 2015.

A. Skariah, P. R, R. R, and B. C. R. Health monitoring of rolling element bearings using improved wavelet cross spectrum technique and support vector machines. *Tribology International*, page 106650, 2020.

J. Q. Smith and P. E. Anderson. Conditional independence and chain event graphs. *Artificial Intelligence*, 172(1):42–68, 2008.

Z. Song, S. Ali, and N. Bouguila. Bayesian inference for infinite asymmetric Gaussian mixture with feature selection. *Soft Computing*, 25:6043–6053, 2021.

A. Soualhi, G. Clerc, H. Razik, and F. Rivas. Long-term prediction of bearing condition by the neo-fuzzy neuron. In *2013 9th IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives (SDEMPED)*, pages 586–591, 2013.

A. Soualhi, H. Razik, G. Clerc, and D. D. Doan. Prognosis of bearing failures using hidden Markov models and the adaptive neuro-fuzzy inference system. *IEEE Transactions on Industrial Electronics*, 61(6):2864–2874, 2014.

P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman. *Causation, Prediction, and Search*. The MIT press, 2000.

N. Stadler and S. Mukherjee. Penalized estimation in high-dimensional hidden Markov models with state-specific graphical models. *The Annals of Applied Statistics*, 7(4):2157–2179, 2013.

I. Stanculescu, C. K. I. Williams, and Y. Freer. Autoregressive hidden Markov models for the early detection of neonatal sepsis. *IEEE Journal of Biomedical and Health Informatics*, 18 (5):1560–1570, 2014.

W. Sun, H. Zhang, A. Palazoglu, A. Singh, W. Zhang, and S. Liu. Prediction of 24-hour-average $PM_{2.5}$ concentrations using a hidden Markov model with different emission distributions in Northern California. *The Science of the Total Environment*, 443C:93–103, 2013.

E. Sutrisno, H. Oh, A. S. S. Vasan, and M. Pecht. Estimation of remaining useful life of ball bearings using data driven methodologies. In *2012 IEEE Conference on Prognostics and Health Management*, pages 1–7. IEEE, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

D. Tobon, K. Medjaher, N. Zerhouni, and G. Tripot. A data-driven failure prognostics method based on mixture of Gaussians hidden Markov models. *IEEE Transactions on Reliability*, 61(2), 2012.

S.-T. Tseng. Optimal preventive maintenance policy for deteriorating production systems. *IIE Transactions*, 28(8):687–694, 1996.

D. G. Tzikas, A. C. Likas, and N. P. Galatsanos. The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146, 2008.

T. Vairo, M. Lecca, E. Trovatore, A. Reverberi, and B. Fabiano. A Bayesian belief network for local air quality forecasting. *Chemical Engineering Transactions*, 74:271–276, 2019.

A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

C. Vitolo, M. Scutari, M. Ghalaieny, A. Tucker, and A. Russell. Modeling air pollution, climate, and health data using Bayesian networks: A case study of the English regions. *Earth and Space Science*, 5(4):76–88, 2018.

L. Wang and H. Shen. Improved data streams classification with fast unsupervised feature selection. In *2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pages 221–226, 2016.

T.-S. Wang, N.-N. Zheng, Y. Li, Y.-Q. Xu, and H.-Y. Shum. Learning kernel-based HMMs for dynamic sequence synthesis. *Graphical Models*, 65(4):206–221, 2003.

X.-D. Wang, R.-C. Chen, F. Yan, and H. Hendry. K-means clustering with feature selection for stream data. In *2018 International Symposium on Computer, Consumer and Control (IS3C)*, pages 453–456, 2018.

X.-T. Wang and X.-Z. Luan. Bayesian penalized method for streaming feature selection. *IEEE Access*, 7:103815–103822, 2019.

Y. Wang and M. Liang. An adaptive SK technique and its application for fault detection of rolling element bearings. *Mechanical Systems and Signal Processing*, 25:1750–1764, 2010.

J. Wu, C. Wu, S. Cao, S. W. Or, C. Deng, and X. Shao. Degradation data-driven time-to-failure prognostics approach for rolling element bearings in electrical machines. *IEEE Transactions on Industrial Electronics*, 66(1):529–539, 2019.

W. Xu, J. Wu, Z. Huang, and C. Guan. Kernel based hidden Markov model with applications to EEG signal classification. In *Proceedings of the 3rd IASTED International Conference on Biomedical Engineering*, pages 401–404, 2005.

R. Yang, F. Yan, and N. Zhao. Urban air quality based on Bayesian network. In *2017 IEEE 9th International Conference on Communication Software and Networks*, pages 1003–1006, 2017.

D. You, R. Li, S. Liang, M. Sun, X. Ou, F. Yuan, L. Shen, and X. Wu. Online causal feature selection for streaming features. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2021.

J. Yu. Adaptive hidden Markov model-based online learning framework for bearing faulty detection and performance degradation monitoring. *Mechanical Systems and Signal Processing*, 83:149–162, 2017.

W. Yue, Y. Wong, and G. Hong. Adaptive-VDHMM for prognostics in tool condition monitoring. In *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*, pages 131–136, 2015.

S. Zhang, B. Guo, A. Dong, Z. Xu, and S. Chen. Cautionary tales on air-quality improvement in Beijing. *Proceedings of the Royal Society A*, 473(2205):20170457, 2017.

C. Zhao and F. Gao. Critical-to-fault-degradation variable analysis and direction extraction for online fault prognostic. *IEEE Transactions on Control Systems Technology*, 25(3): 842–854, 2017.

Y. Zheng, B. Jeon, L. Sun, J. Zhang, and H. Zhang. Student's t-hidden Markov model for unsupervised learning using localized feature selection. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2586–2598, 2018.

H. Zhou, M. You, L. Liu, and C. Zhuang. Sequential data feature selection for human motion recognition via Markov blanket. *Pattern Recognition Letters*, 86:18–25, 2017.

P. Zhou, X. Hu, P. Li, and X. Wu. Online streaming feature selection using adapted neighborhood rough set. *Information Sciences*, 481:258–279, 2019a.

P. Zhou, X. Hu, P. Li, and X. Wu. Ofs-density: A novel online streaming feature selection method. *Pattern Recognition*, 86:48–61, 2019b.

P. Zhou, N. Wang, and S. Zhao. Online group streaming feature selection considering feature interaction. *Knowledge-Based Systems*, 226:107157, 2021.

H. Zhu, Z. He, and H. Leung. Simultaneous feature and model selection for continuous hidden Markov models. *IEEE Signal Processing Letters*, 19(5):279–282, 2012.

Appendix A

Supplementary material for AR-AsLG-HMMs

A.1 Parameters used in the synthetic data

The parameters of the two scenarios for the synthetic data can be seen in Table A.1 and Table A.2. In particular, in Table A.1 and Table A.2 the parameters for the AR-AsLG-HMM and AR-MoG-HMM emission probabilities are written respectively.

Par	Value	Par	Value	Par	Value
φ_{11}^t	1.5	f_{21}^t	$1.5 + 0.2x_1^{t-1}$	φ_{31}^t	$1.5 + .999x_1^{t-1}$
φ_{12}^t	2.5	f_{22}^t	$2.5x_3^t$	φ_{32}^t	$2.5 + 5.0x_1^t + 8.0x_4^t + .888x_2^{t-1} + .111x_2^{t-2}$
φ_{13}^t	4.5.	f_{23}^t	$3.0x_5^t + .99x_3^{t-1}$	φ_{33}^t	$4.5 + 1.5x_1^t + .999x_3^{t-1}$
φ_{14}^t	3.5	f_{24}^t	$1.5 + 9.5x_1^t$	φ_{34}^t	$3.5 + 1.5x_1^t + 2.0x_3^t + .1x_4^{t-1}$
φ_{15}^t	6.5	f_{25}^t	$6.5 + .99x_5^{t-1}$	φ_{35}^t	$5.0x_3^t$
φ_{16}^t	1.5	f_{26}^t	$.5 + 6.8x_5^t$	φ_{36}^t	$1 + 3.5x_3^t - 4.5x_5^t + .8x_6^{t-1}$
σ_{11}	1.5	σ_{21}	3.5	σ_{31}	3
σ_{12}	2.0	σ_{22}	2	σ_{32}	3.5
σ_{13}	3.0	σ_{23}	4	σ_{33}	4.0
σ_{14}	8.0	σ_{24}	2	σ_{34}	6.5
σ_{15}	6.0	σ_{25}	5.5	σ_{35}	5.5
σ_{16}	.5	σ_{26}	2	σ_{36}	7.0

Table A.1: Scenario 1 parameters of AR-AsLG-HMM distribution

Let $\boldsymbol{D}$ and $\boldsymbol{L}$ be the matrices:

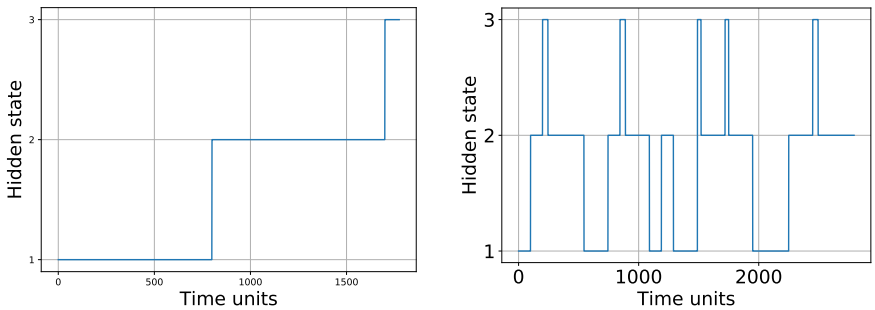
$$\boldsymbol{D} = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}, \quad \boldsymbol{L} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0.3 & 0 & 0 & 0 & 0 \\ -0.3 & 0.1 & 0.2 & 0 & 0 & 0 \\ 0.6 & 0.1 & 0.2 & -0.3 & 0 & 0 \\ 0.1 & -0.2 & 0.4 & 0.5 & 0.3 & 0 \end{bmatrix}$$

Par	Value	Par	Value	Par	Value
μ^t_{111}	$1 + .2x_1^{t-1}$	μ^t_{121}	$1 + .8x_1^{t-1}$	μ^t_{211}	$3 + .8x_1^{t-1} + .1x_1^{t-2}$
μ^t_{112}	$3 + .9x_2^{t-1}$	μ^t_{122}	$3 + .1x_2^{t-1}$	μ^t_{212}	$6 + .3x_2^{t-1} + .1x_2^{t-2}$
μ^t_{113}	$1 + .1x_3^{t-1}$	μ^t_{123}	$1 + .5x_3^{t-1} + .2x_3^{t-2}$	μ^t_{213}	$4 + .6x_3^{t-1}$
μ^t_{114}	$4 - .1x_4^{t-1}$	μ^t_{124}	$4 - .9x_4^{t-1} + .3x_4^{t-2}$	μ^t_{214}	$6 + .2x_4^{t-1}$
μ^t_{115}	1	μ^t_{125}	$1 + .9x_5^{t-1}$	μ^t_{215}	$4 + .1x_5^{t-1}$
μ^t_{116}	$1 + .9x_6^{t-1}$	μ^t_{126}	$1 + .2x_6^{t-1}$	μ^t_{216}	$3 + .8x_6^{t-1}$
Σ_{11}	$.2\boldsymbol{D} + .2(\boldsymbol{L} + \boldsymbol{L}')$	Σ_{12}	$.4\boldsymbol{D} + .3(\boldsymbol{L} + \boldsymbol{L}')$	Σ_{21}	$.2\boldsymbol{D} + .2(\boldsymbol{L} + \boldsymbol{L}')$
μ^t_{221}	$3 + 0.1x_1^{t-1}$	μ^t_{311}	$10 + .6x_1^{t-1} + 0.2x_1^{t-2}$	μ^t_{321}	$10 + .1x_1^{t-1} + .3x_1^{t-2}$
μ^t_{222}	$6 + .6x_2^{t-1} + .3x_2^{t-2}$	μ^t_{312}	$9 + .99x_2^{t-1}$	μ^t_{322}	$9.3 + .99x_2^{t-1}$
μ^t_{223}	$4 + .8x_3^{t-1} + .1x_3^{t-2}$	μ^t_{313}	$12 + .1x_3^{t-1}$	μ^t_{323}	$12 + .6x_3^{t-1} + .1x_3^{t-2}$
μ^t_{224}	$6 + .5x_4^{t-1} + .2x_4^{t-2}$	μ^t_{314}	$13 + .1x_4^{t-1}$	μ^t_{324}	$13 + .8x_4^{t-1}$
μ^t_{225}	$4 + .6x_5^{t-1}$	μ^t_{315}	$5 + .999x_5^{t-1}$	μ^t_{325}	$5 + .999x_5^{t-1}$
μ^t_{226}	$3 + .2x_6^{t-1}$	μ^t_{316}	$15 + .8x_6^{t-1} + 0.05x_6^{t-2}$	μ^t_{326}	$15 + .2x_6^{t-1} + .1x_6^{t-2}$
Σ_{22}	$.3\boldsymbol{D} - .3(\boldsymbol{L} + \boldsymbol{L}')$	Σ_{31}	$.2\boldsymbol{D} - .2(\boldsymbol{L} + \boldsymbol{L}')$	Σ_{32}	$.5\boldsymbol{D} - .3(\boldsymbol{L} + \boldsymbol{L}')$

Table A.2: Scenario 2 parameters of AR-MoG-HMM distribution

$\boldsymbol{\Sigma} = a\boldsymbol{D} + b(\boldsymbol{L} + \boldsymbol{L}')$ is a Hermitian matrix with strict dominant diagonal where $\boldsymbol{L}'$ is the transpose of $\boldsymbol{L}$, $|a| > |b| > 0$ and $|b| < 1$. The parameters a, $b \in \mathcal{R}$ are tuned to obtain matrices with positive eigenvalues and therefore definite positive matrices. For all the hidden states, the mixtures have weights $w_1 = 0.7$ and $w_2 = 0.3$. In Table A.2, parameter μ_{ijk} stands for the mean of the k-variable at the j-mixture component of the i-hidden state and $\boldsymbol{\Sigma}_{ij}$ is the covariance matrix of the j-mixture component of the i-hidden state.

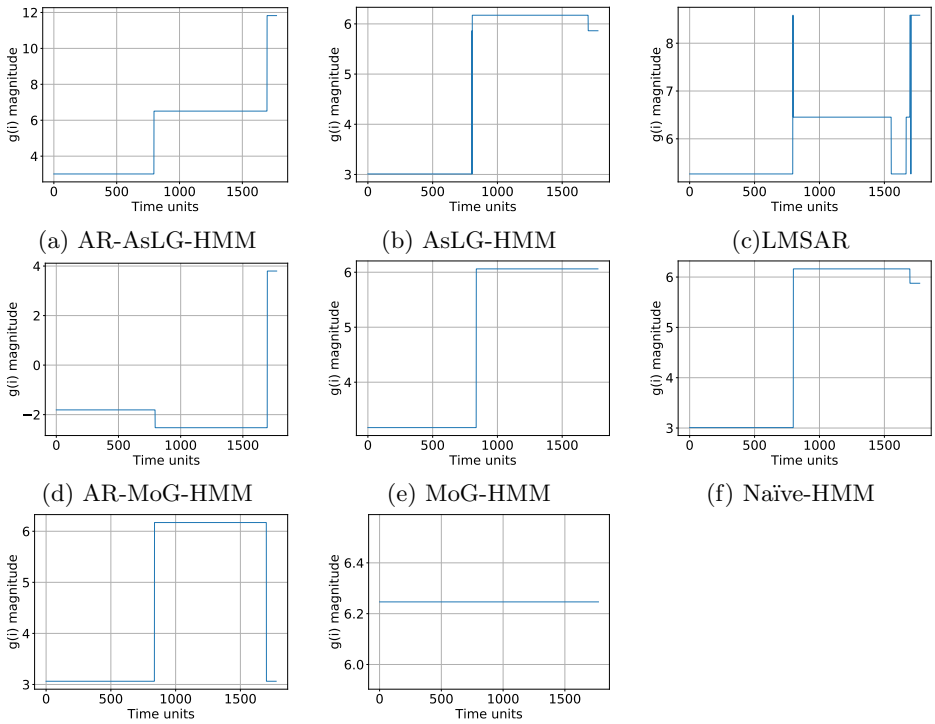
A.2 Viterbi paths for synthetic data



(a) Sequence 1 (b) Sequence 2

Figure A.1: Sequences of hidden states used to construct the test signals. Sequence 1 (a) and sequence 2 (b) are used for both scenarios

Fig. A.2 and Fig. A.3 show the Viterbi paths in log scale obtained for sequences 1 and 2 for scenario 1. These paths are interesting since they convey the changes in the dynamics of the data for every time instance (smoothing). When new instances arrive, the Viterbi algorithm can be used to determine the hidden state to which the new instance belongs (filtering) and make decisions or analyses of the observed process. In this case, observe that AR-AsLG-HMM obtained a good Viterbi paths that follow the pattern described by Fig. 5.3 (a) and (b). On the other hand, AR-MoG-HMM, AsLG-HMM and naïve-HMM obtained fair Viterbi



(a) AR-AsLG-HMM (b) AsLG-HMM (c)LMSAR

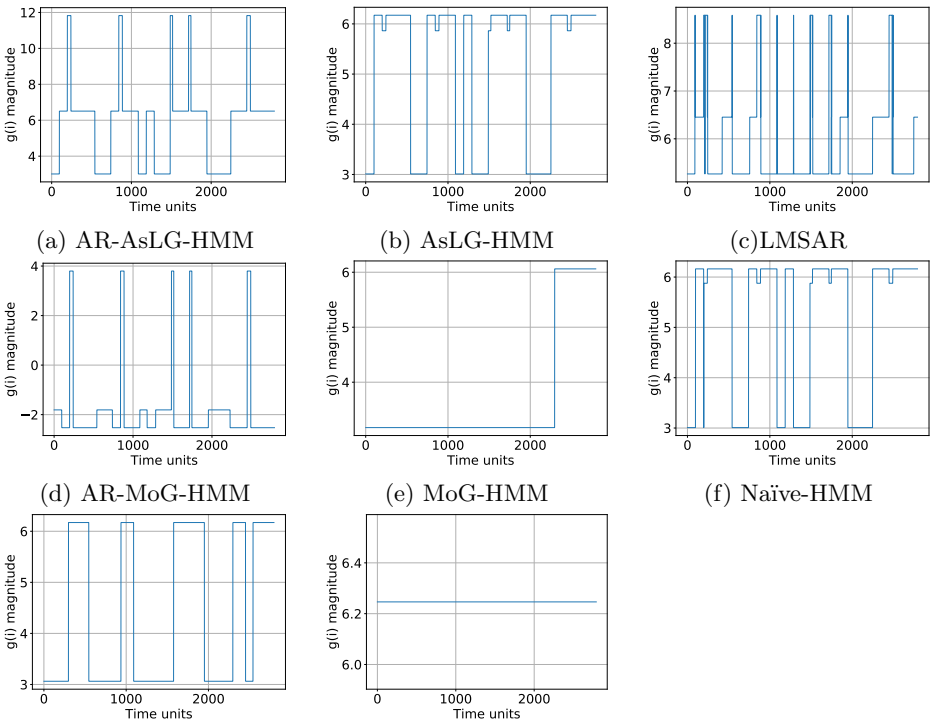
(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM

Figure A.2: Viterbi paths for scenario 1 and sequence 1

path, since the transitions are correct but wrong in magnitude. LMSAR obtained a fair result for sequence 1 but the transitions are wrongly predicted. It is relevant to observe that AsLG-HMM and naïve-HMM obtained similar results but far different from those obtained by AR-AsLG-HMM. This indicates the relevance of assuming AR parameters when relevant AR parameters are present in the data. The remaining models obtained poor results.

Fig. A.4 and Fig. A.5 show the Viterbi paths obtained for sequences 1 and 2 for scenario 2. The Viterbi paths obtained by asymmetric models and naïve-HMM were good, since they follow correctly the pattern of the hidden state sequences shown in Fig. 5.3. The possible reason behind this is that the autoregressive processes in this case were not relevant for the data generation. This can be seen clearer when observing the similar $g_1(i)$ values among some models, e.g., for AR-AsLG-HMM, for both sequences, the biggest $g_1(i)$ value was 400.98, for AsLG-HMM was 398.71 and for naïve-HMM was 406.037. On the other hand, AR-MoG-HMM and BMM obtained fair results since they follow the state transitions but with wrong amplitude. In particular, in spite that AR-MoG-HMM and BMM obtain similar results, it must be noticed the differences in amplitude in their predictions: AR-MoG-HMM records amplitudes of 6 units in $g_1(i)$ and BMM shows amplitudes of 4000 units in $g_1(i)$. Meanwhile, VAR-MVGHMM had problems with the state durations. Finally, note that the remaining models obtained not well-defined readings of the evolution of hidden states proposed by Fig. 5.3.

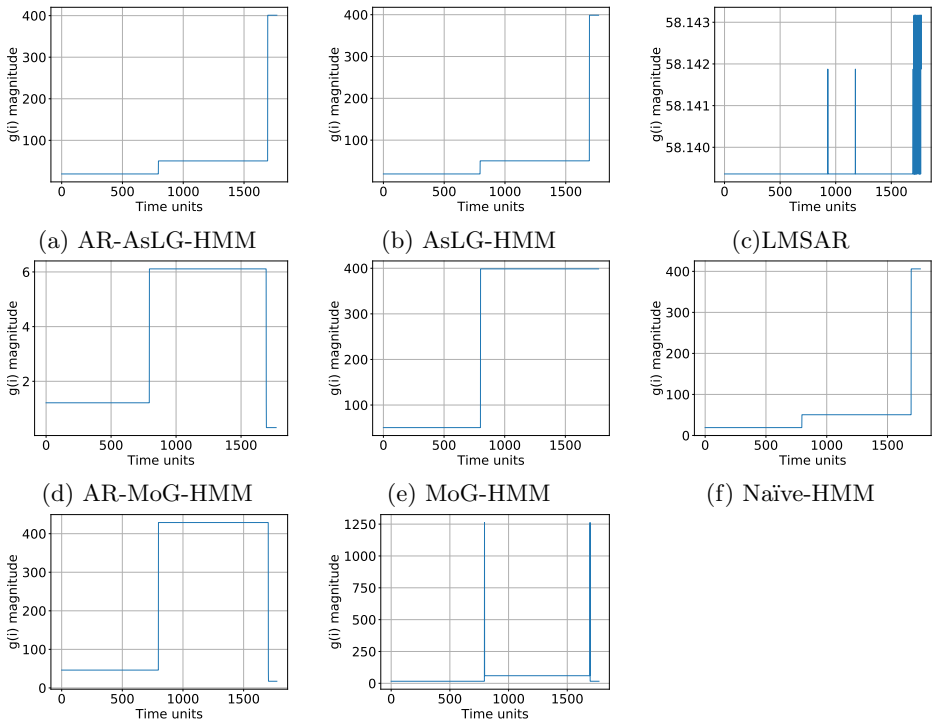


(a) AR-AsLG-HMM (b) AsLG-HMM (c)LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM

Figure A.3: Viterbi paths for scenario 1 and sequence 2



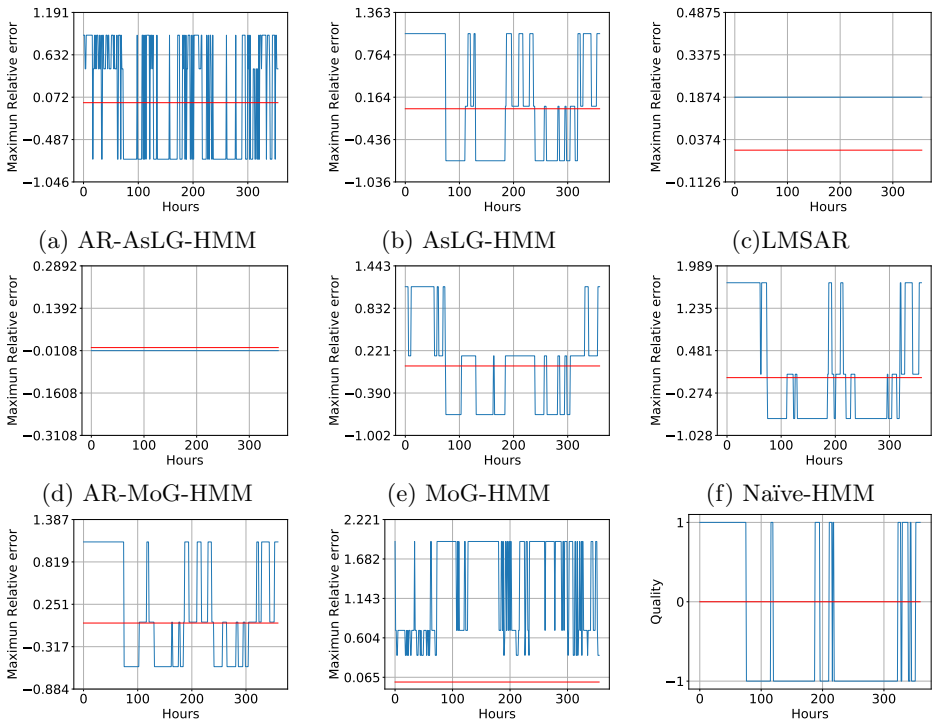
(a) AR-AsLG-HMM (b) AsLG-HMM (c)LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM

Figure A.4: Viterbi paths for scenario 2 and sequence 1

A.3 Viterbi path for air quality when three hidden states are used

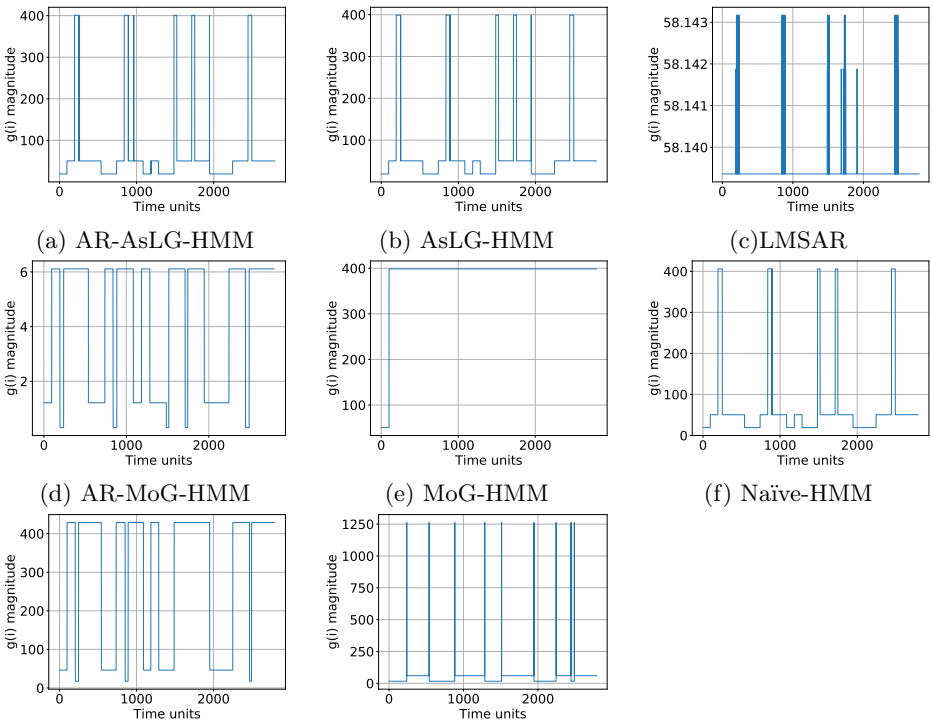


(a) AR-AsLG-HMM (b) AsLG-HMM (c)LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM (i) Real

Figure A.6: Viterbi paths for the air quality example during the first week of 2016 when three hidden states are used



(a) AR-AsLG-HMM (b) AsLG-HMM (c)LMSAR

(d) AR-MoG-HMM (e) MoG-HMM (f) Naïve-HMM

(g) BMM (h) VAR-MVGHMM

Figure A.5: Viterbi paths for scenario 2 and sequence 2

Fig. A.6 shows the predicted air quality for the first two weeks of 2016 for each model using the Viterbi algorithm when three hidden states are assumed. In this scenario, it is still observable the same details when two hidden states are assumed. Nevertheless, in the BIC and LL scores, the models obtained better results.

A.4 Proofs to lemmas and theorems

Lemma A.1. Let $\boldsymbol{\lambda}^{(s)}$ be the parameters at iteration s of the EM and $\boldsymbol{\lambda}^{(s+1)}$ be the resulting parameters after the next iteration of the EM. The following follows: $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) \geq \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)})$.

■

Proof. From the maximization step $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}^{(s)}) = \max_{\boldsymbol{\lambda}'} \mathcal{Q}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}^{(s)}) \geq \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)})$. □

Lemma A.2. Given two arbitrary models with respective parameters $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$, the following follows $\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') \leq \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}')$, and the equality holds when $P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}) = P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}')$.

■

Proof. Notice that:

$$\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') - \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}') = \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}') \ln \frac{P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda})}{P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}')}. \tag{A.1}$$

Observe that if $P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}) = P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}')$, then the logarithm in Eq. (**??**) is zero and $\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') = \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}')$. However, if the Jensen's inequality is applied for concave functions to Eq. (A.1). It holds:

$$\begin{aligned}\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') - \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}') &\leq \ln \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}') \frac{P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda})}{P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}')} \\ &= \ln \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}).\end{aligned}$$

However $\ln \sum_{R(\boldsymbol{Q}^{p^*:T})} P(\boldsymbol{q}^{p^*:T}|\boldsymbol{x}^{0:T}, \boldsymbol{\lambda}) \leq 0$, and therefore $\mathcal{H}^{p^*}(\boldsymbol{\lambda}|\boldsymbol{\lambda}') \leq \mathcal{H}^{p^*}(\boldsymbol{\lambda}'|\boldsymbol{\lambda}')$ as desired. □

Theorem A.1. Let $\boldsymbol{\lambda}^{(s)}$ be the parameters at an iteration s of the EM and $\boldsymbol{\lambda}^{(s+1)}$ be the resulting parameters after the next iteration of the EM. The following holds:

(a) $LL(\boldsymbol{\lambda}^{(s+1)}) \geq LL(\boldsymbol{\lambda}^{(s)})$. In other words, the log-likelihood of the model cannot worsen after an EM iteration.

(b) The sequence $\{LL(\boldsymbol{\lambda}^{(s)})\}_{s\in\mathbb{N}}$ converges.

■

Proof. To prove (a), the following identity follows: $LL(\boldsymbol{\lambda}^{(s+1)})) = \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) - \mathcal{H}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)})$. Note that:

$$LL(\boldsymbol{\lambda}^{(s+1)}) - LL(\boldsymbol{\lambda}^{(s)}) = \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) - \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)}) + \mathcal{H}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)}) - \mathcal{H}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}).$$

From Lemma 1, it holds: $\mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) - \mathcal{Q}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)}) \geq 0$ and from Lemma 2, then, $\mathcal{H}^{p^*}(\boldsymbol{\lambda}^{(s)}|\boldsymbol{\lambda}^{(s)}) - \mathcal{H}^{p^*}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)}) \geq 0$. Therefore $LL(\boldsymbol{\lambda}^{(s+1)}) - LL(\boldsymbol{\lambda}^{(s)}) \geq 0$ and the desired results are obtained.

To prove (b), from (a) it is known that the sequence $\{LL(\boldsymbol{\lambda}^{(s)})\}_{s\in Z^+}$ does not decrease and is also upper bounded by zero. Therefore, $\{LL(\boldsymbol{\lambda}^{(s)})\}_s$ converges to a certain real finite number LL^* with $LL^* \leq 0$. □

Lemma A.3. $\alpha^t_{p^*}(i)$ and $\beta^t_{p^*}(i)$ can be computed as:

$$\begin{aligned}\alpha^t_{p^*}(i) &= \sum_{j=1}^{N} b_i^{p^*}(\boldsymbol{x}^t) a_{ji} \alpha^{t-1}(j) \\ \beta^t_{p^*}(i) &= \sum_{j=1}^{N} \beta^{t+1}(j) b_j^{p^*}(\boldsymbol{x}^{t+1}) a_{ij}\end{aligned} \tag{A.2}$$

for $t = p^*, ..., T$ and $i = 1, ..., N$, with initial values $\alpha_{p^*}^{p^*}(i) = \pi_i b_i^{p^*}(\boldsymbol{x}^{p^*})$ and $\beta_{p^*}^{T}(i) = 1$, $i = 1, ..., N$.

■

Proof. For the forward variable, for $t = p^* + 1, ..., T$ and $i = 1, ..., N$:

$$\begin{aligned}
\alpha_{p^*}^{t}(i) &= \sum_{j=1}^{N} P(Q^t = i, Q^{t-1} = j, \boldsymbol{x}^{p^*:t} | \boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^t | Q^t = i, \boldsymbol{x}^{t-p^*:t-1}, \boldsymbol{\lambda}) P(Q^t = i, Q^{t-1} = j, \boldsymbol{x}^{p^*:t-1} | \boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} b_i^{p^*}(\boldsymbol{x}^t) P(Q^t = i | Q^{t-1} = j, \boldsymbol{\lambda}) P(Q^{t-1} = j, \boldsymbol{x}^{p^*:t-1} | \boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} b_i^{p^*}(\boldsymbol{x}^t) a_{ji} \alpha_{p^*}^{t-1}(j).
\end{aligned} \tag{A.3}$$

In the second equality of Eq. (A.3), note that $\boldsymbol{X}^t$ is D-separated from Q^{t-1} given Q^t and $\boldsymbol{X}^{t-p^*:t-1}$. In the third equality, observe that Q^t is D-separated from $\boldsymbol{X}^{0:t-1}$ given Q^{t-1}. D-separation implies conditional independence in Bayesian networks. Therefore, the forward variable can be computed iteratively as in the traditional HMM. Additionally, the forward variable is initialized with $\alpha_{p^*}^{p^*}(i) = \pi_i b_i^{p^*}(\boldsymbol{x}^{p^*})$, $i = 1, ..., N$.

In the case of the backward variable, for $t = T - 1, ..., p^*$ and $i = 1, ..., N$:

$$\begin{aligned}
\beta_{p^*}^{t}(i) &= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+1:T}, Q^{t+1} = j | Q^t = i, \boldsymbol{x}^{0:t}, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} P(\boldsymbol{x}^{t+2:T} | Q^{t+1} = j, \boldsymbol{x}^{0:t+1}, \boldsymbol{\lambda}) P(\boldsymbol{x}^{t+1}, Q^{t+1} = j | Q^t = i, \boldsymbol{x}^{0:t}, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} \beta_{p^*}^{t+1}(j) P(\boldsymbol{x}^{t+1} | Q^{t+1} = j, \boldsymbol{x}^{t+1-p^*:t}, \boldsymbol{\lambda}) P(Q^{t+1} = j | Q^t = i, \boldsymbol{\lambda}) \\
&= \sum_{j=1}^{N} \beta_{p^*}^{t+1}(j) b_j^{p^*}(\boldsymbol{x}^{t+1}) a_{ij}.
\end{aligned} \tag{A.4}$$

In the second equality of Eq. (A.4), the D-separation is again applied, specifically $\boldsymbol{X}^{t+2:T}$ is D-separated from Q^t given Q^{t+1} and $\boldsymbol{X}^{0:t+1}$. In the third equality, $\boldsymbol{X}^{t+1}$ is D-separated from Q^t given Q^{t+1} and $\boldsymbol{X}^{0:t}$. Additionally, Q^{t+1} is D-separated from $\boldsymbol{X}^{0:t}$ given Q^t; also, $\boldsymbol{X}^{t+1}$ is D-separated of $\boldsymbol{X}^{0:t-p^*}$ given $\boldsymbol{X}^{t+1-p^*:t}$ because each $\boldsymbol{X}^t$ is dependent on maximum p^* lags. Hence, the backward variable can be computed iteratively as in the traditional HMM. Finally, the backward variable is initialized with $\beta_{p^*}^{T}(i) = 1$, $i = 1, ..., N$. □

Theorem A.2. The M-step for an AR-AsLG-HMM model can be performed using the

following updating formulas: parameter $\boldsymbol{\pi} = \{\pi_i\}_{i=0}^N$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{A.5}$$

The parameter $\mathbf{A} = \{a_{ij}\}_{i,j=1}^N$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{A.6}$$

If $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$, the parameters $\{\eta_{imr}\}_{r=1}^{p_{im}}$, $\{\beta_{imk}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear system:

$$\begin{cases} \sum_{t=p^*}^T \gamma^t(i) x_m^t = \sum_{t=p^*}^T \gamma^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^T \gamma^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^T \gamma^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^T \gamma^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^T \gamma^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^T \gamma^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^T \gamma^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^T \gamma^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^T \gamma^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{A.7}$$

if $\boldsymbol{\theta}_{im} = (\boldsymbol{\beta}_{im} | \boldsymbol{\eta}_{im})^\top$, $\boldsymbol{o}_{im}^t = (\boldsymbol{u}_{im}^t | \boldsymbol{d}_{im}^t)$, and $\boldsymbol{\Gamma}_i^{p^*:T} := \text{Matrix}([\gamma^{p^*}(i), ..., \gamma^T(i)])$ Then, the previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{A.8}$$

If $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$, then, σ_{im}^2 can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^T \gamma^t(i)(x_m^t - \hat{\varphi}_{im}^t)^2}{\sum_{t=p^*}^T \gamma^t(i)}. \tag{A.9}$$

This update must be done for every variable $m = 1, ..., M$ and hidden state $i = 1, ..., N$.

■

Proof. The Lagrange multipliers are applied with the restrictions $\sum_{i=1}^N \pi_i = 1$ and $\sum_{j=1}^N a_{ij} = 1$ for $i = 1, ..., N$. The corresponding Lagrangian function is:

$$\mathcal{L}(\boldsymbol{\lambda}, \tau_0, \tau_1, ..., \tau_N) = \mathcal{Q}^{p^*}(\boldsymbol{\lambda} | \boldsymbol{\lambda}') + \tau_0 (1 - \sum_{i=1}^N \pi_i) + \sum_{i=1}^N \tau_i (1 - \sum_{j=1}^N a_{ij}). \tag{A.10}$$

The derivative of $\mathcal{L}$ in Eq. (A.10) with respect to π_i and equalized to zero gives as result:

$$\frac{\partial \mathcal{L}}{\partial \pi_i} = \frac{\gamma^{p^*}(i)}{\pi_i} - \tau_0 = 0. \tag{A.11}$$

Then, $\pi_i = \frac{\gamma^{p^*}(i)}{\tau_0}$, and $\sum_{i=1}^{N} \pi_i = \frac{\sum_{i=1}^{N} \gamma^{p^*}(i)}{\tau_0}$. Hence, $\tau_0 = 1$. Therefore, the updating formula for π_i, for $i = 1, ..., N$, is:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{A.12}$$

Similarly, with respect to a_{ij}:

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = \sum_{t=p^*}^{T-1} \frac{\xi^t(i,j)}{a_{ij}} - \tau_i = 0. \tag{A.13}$$

Then, $a_{ij} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\tau_i}$, adding over all the hidden states values, it holds that:

$$\sum_{j=1}^{N} a_{ij} = \frac{\sum_{t=p^*}^{T-1} \sum_{j=1}^{N} \xi^t(i,j)}{\tau_i} = \frac{\sum_{t=p^*}^{T-1} \gamma^t(i)}{\tau_i} = 1. \tag{A.14}$$

From the previous equation, $\tau_i = \sum_{t=p^*}^{T-1} \gamma^t(i)$. And the updating formula for a_{ij} for $i, j = 1, ..., N$ is:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{A.15}$$

Now, the derivative of $\mathcal{L}$ in Eq. (A.10) are computed, with respect to the parameters η_{imp}, β_{imk} and σ_{im}^2. For the derivative with respect to β_{im0}:

$$\frac{\partial \mathcal{L}}{\partial \beta_{im0}} = \sum_{t=p^*}^{T} \gamma^t(i) \frac{\partial}{\partial \beta_{im0}} \ln(\mathcal{N}(x_m^t | \varphi_{im}^t, \sigma_{im}^2)). \tag{A.16}$$

Thus,

$$0 = \sum_{t=p^*}^{T} \frac{\gamma^t(i)}{\sigma_{im}^2} (\varphi_{im}^t - x_m^t).$$

Then,

$$\sum_{t=p^*}^{T} \gamma^t(i) x_m^t = \sum_{t=p^*}^{T} \gamma^t(i) \varphi_{im}^t. \tag{A.17}$$

Now, if $\mathcal{L}$ in Eq. (A.10) is derived with respect to β_{imk}, with $k = 1, ..., k_{im}$, and with respect

to η_{imr} with $r = 1, ..., p_{im}$ as in Eq. (A.16), the following equations are obtained:

$$\begin{aligned}
\sum_{t=p^*}^{T} \gamma^t(i) x_m^t u_{im1}^t &= \sum_{t=p^*}^{T} \gamma^t(i) u_{im1}^t \varphi_{im}^t \\
\sum_{t=p^*}^{T} \gamma^t(i) x_m^t u_{im2}^t &= \sum_{t=p^*}^{T} \gamma^t(i) u_{im2}^t \varphi_{im}^t \\
\vdots \qquad & \vdots \qquad\qquad \vdots \\
\sum_{t=p^*}^{T} \gamma^t(i) x_m^t u_{imk_{im}}^t &= \sum_{t=p^*}^{T} \gamma^t(i) u_{imk_{im}}^t \varphi_{im}^t \\
\sum_{t=p^*}^{T} \gamma^t(i) x_m^t x_m^{t-1} &= \sum_{t=p^*}^{T} \gamma^t(i) x_m^{t-1} \varphi_{im}^t \\
\vdots \qquad & \vdots \qquad\qquad \vdots \\
\sum_{t=p^*}^{T} \gamma^t(i) x_m^t x_m^{t-p_{im}} &= \sum_{t=p^*}^{T} \gamma^t(i) x_m^{t-p_{im}} \varphi_{im}^t
\end{aligned} \tag{A.18}$$

The previous linear system can be written and solved as:

$$\begin{aligned}
(\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{x}_m^{p^*:T} &= (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \boldsymbol{\theta}_{im} \\
\boldsymbol{\theta}_{im}^{(s+1)} &= \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Gamma}_i^{p^*:T} \boldsymbol{x}_m^{p^*:T}
\end{aligned} \tag{A.19}$$

The solution to this system of equations returns the coefficients $\{\beta_{imk}^{(s+1)}, \eta_{iml}^{(s+1)}\}_{k=0,l=1}^{k_{im},p_{im}}$, for each variable X_m, $m = 1, 2, ..., M$ and hidden state $i \in R(Q)$. Once these parameters are known, the mean $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$ can be computed. To update σ_{im}^2, the derivative of $\mathcal{L}$ in Eq. (A.10) with respect to σ_{im}^2 is computed and equalize to zero:

$$\frac{\partial \mathcal{L}}{\partial \sigma_{im}^2} = \sum_{t=p^*}^{T} \gamma^t(i) \frac{\partial}{\partial \sigma_{im}^2} \ln(\mathcal{N}\big(x_m^t | \hat{\varphi}_{im}^t, \sigma_{im}^2\big)).$$

Thus

$$0 = \sum_{t=p^*}^{T} \gamma^t(i) \big(\frac{(x_m^t - \hat{\varphi}_{im}^t)^2}{\sigma_{im}^4} - \frac{1}{\sigma_{im}^2} \big).$$

Hence,

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^{T} \gamma^t(i)(x_m^t - \hat{\varphi}_{im}^t)^2}{\sum_{t=p^*}^{T} \gamma^t(i)}. \tag{A.20}$$

□

Lemma A.4. If $\delta^t_{p^*}(i) = \max_{\boldsymbol{q}^{p^*:t-1}}\{P(\boldsymbol{x}^{p^*:t}, \boldsymbol{q}^{p^*:t-1}, Q^t = i|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})\}$ represents the most probable sequence of hidden states up to time $t-1$ for state i at time t, then $\delta^t_{p^*}(i)$ can be computed recursively.

$$\delta^t_{p^*}(i) = \max_{j=1,\dots,N}\{\delta^{t-1}_{p^*}(j)a_{ji}\}b^{p^*}_i(\boldsymbol{x^t})$$

The Viterbi algorithm is initialized with $\delta^{p^*}_{p^*}(i) = \pi_i b^{p^*}_i(\boldsymbol{x}^{p^*})$.

■

Proof. Observe that:

$$\begin{aligned}
\delta^t_{p^*}(i) &= \max_{\boldsymbol{q}^{p^*:t-1}}\{P(\boldsymbol{x}^t|\boldsymbol{x}^{t-p^*:t-1}, Q^t = i, \boldsymbol{\lambda})P(\boldsymbol{x}^{p^*:t-1}, \boldsymbol{q}^{p^*:t-1}, Q^t = i|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})\} \\
&= \max_{\boldsymbol{q}^{p^*:t-1}}\{b^{p^*}_i(\boldsymbol{x^t})P(Q^t = i|q^{t-1}, \boldsymbol{\lambda})P(\boldsymbol{x}^{p^*:t-1}, \boldsymbol{q}^{p^*:t-1}|\boldsymbol{x}^{0:p^*-1}, \boldsymbol{\lambda})\} \\
&= \max_{\boldsymbol{q}^{p^*:t-1}}\{b^{p^*}_i(\boldsymbol{x^t})a_{q^{t-1}i}\delta^{t-1}_{p^*}(q^{t-1})\} \\
&= \max_{j=1,\dots,N}\{\delta^{t-1}_{p^*}(j)a_{ji}\}b^{p^*}_i(\boldsymbol{x^t}).
\end{aligned} \tag{A.21}$$

For the first equality of Eq. (A.21), notice that $\boldsymbol{X}^t$ is D-separated of $Q^{p^*:t-1}$ given Q^t and $\boldsymbol{X}^{t-p^*:t-1}$. In the second equality, as Q^t is being D-separated from $\boldsymbol{X}^{0:t-1}$ and $\boldsymbol{Q}^{p^*:t-2}$ given Q^{t-1}. In the third equality, the dynamic programming principle Forney [1973], Omura [1969] is applied in $\delta^{t-1}_{p^*}$. Note that, $\delta^t_{p^*}(i)$ can be computed iteratively as in its traditional version. The Viterbi algorithm is initialized with $\delta^{p^*}_{p^*}(i) = \pi_i b^{p^*}_i(\boldsymbol{x}^{p^*})$ for $i = 1, \dots, N$. □

Appendix B

Supplementary material for FS-AsHMM

B.1 Parameters used in the synthetic data

i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}
1	1	1.0	1.2	2	1	2.0	1.5	3	1	3.0	1.4	4	1	4.0	0.8
1	2	2.0	1.3	2	2	4.0	1.7	3	2	6.0	2.2	4	2	8.0	1.1
1	3	1.5	1.0	2	3	1.5	1.0	3	3	1.5	1.0	4	3	1.5	1.0
1	4	0.5	1.3	2	4	1.0	1.2	3	4	1.5	1.7	4	4	2.0	0.9
1	5	4.5	2.2	2	5	4.5	2.2	3	5	4.5	2.2	4	5	4.5	2.2
1	6	1.0	1.4	2	6	3.0	0.7	3	6	6.0	1.6	4	6	9.0	1.3
1	7	2.0	1.3	2	7	3.5	1.3	3	7	4.0	1.3	4	7	6.0	1.3
1	8	0.0	1.0	2	8	1.5	1.8	3	8	4.0	0.5	4	8	7.0	1.4
1	9	1.2	1.5	2	9	2.0	2.0	3	9	2.6	0.8	4	9	3.4	1.2
1	10	3.0	0.6	2	10	3.0	0.6	3	10	3.0	0.6	4	10	3.0	0.6

Table B.1: Scenario1 parameters for the synthetic data

The parameters of the means and standard deviations of the three scenarios for the synthetic data used in Chapter 6 are shown in Table B.1, Table B.2 and Table B.3. Recall that $\varphi_{im} = \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ refers to the mean function of the variable X_m at the hidden state $Q^t = i$. Note that Table B.1 has the data with the less complexity regarding probabilistic relationships between variables; whereas Table B.3 exhibits the most complex dynamical probabilistic behavior among the three possible scenarios. Also, it can be seen that some variables change their parameters for all the hidden states, in such case, the variables are assumed to be relevant. Whereas for others is not the case, and they have constant parameters for all the hidden states, in such cases, the variables are irrelevant or noise. Finally, variables whose parameters change for some hidden states but not all, are considered partially relevant variables.

i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}
1	1	1.0	1.2	2	1	1.5	1.5
1	2	$1.3 + 2.6x_1^t$	1.3	2	2	$3.3 + 3.1x_1^t$	1.7
1	3	1.5	1.0	2	3	1.5	1.0
1	4	0.0	1.3	2	4	$1.0 + 3.2x_2^t$	1.2
1	5	1.5	2.2	2	5	1.5	2.2
1	6	$0.1 + 3.8x_4^t$	1.4	2	6	$0.2 + 3.2x_4^t$	0.7
1	7	$0.6 + 2.7x_8^t$	1.3	2	7	$1.0 + 2.3x_6^t$	1.3
1	8	0.7	1.0	2	8	$3.0 + 1.2x_9^t$	1.8
1	9	$0.2 + 1.2x_1^t$	1.5	2	9	$1.0 + 2.0x_1^t$	2.0
1	10	1.0	0.6	2	10	1.0	0.6
3	1	2.0	1.4	3	1	3.0	0.8
3	2	5.6	2.2	3	2	6.3	1.1
3	3	1.5	1.0	3	3	1.5	1.0
3	4	4.2	1.7	3	4	6.3	0.9
3	5	1.5	2.2	3	5	1.5	2.2
3	6	$0.3 + 7.2x_2^t + 5.2x_4^t$	1.6	3	6	$0.4 + 11.2x_1^t + 15.2x_2^t + 9.2x_4^t$	1.3
3	7	$2.5 + 3.2x_6^t + 5.6x_8^t$	1.3	3	7	$6.2 + 10.1x_1^t + 12.4x_2^t + 14.5x_4^t$	1.3
3	8	$5.0 + 2.2x_1^t + 3.4x_9^t$	0.5	3	8	$6.0 + 6.8x_1^t + 9.1x_2^t + 5.8x_4^t$	1.4
3	9	2.0	0.8	3	9	$4.5 + 6.2x_1^t + 7.4x_2^t + 2.5x_4^t$	1.2
3	10	1.0	0.6	3	10	1.0	0.6

Table B.2: Scenario2 parameters for the synthetic data

i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}
1	1	0.1	1.2	2	1	$0.3 + 0.7x_1^{t-1}$	1.5
1	2	0.5	1.3	2	2	$2.1 + 0.9x_2^{t-1}$	1.7
1	3	1.5	1.0	2	3	1.5	1.0
1	4	0.0	1.3	2	4	$1.2 + 4x_2^t$	1.2
1	5	4.5	2.2	2	5	6.5	2.2
1	6	0.1	1.4	2	6	0.2	0.7
1	7	1.0	1.3	2	7	0.5	1.3
1	8	1.2	1.0	2	8	$3.6 + 2x_6^t + 0.6x_8^{t-1}$	1.8
1	9	0.3	1.5	2	9	$1.2 + 2x_1^t$	2.0
1	10	3.0	0.6	2	10	3.0	0.6
3	1	0.5	1.4	3	1	$1.2 + 12.2x_6^t + 8.2x_8^t + 6.2x_9^t$	0.8
3	2	$3.6 + 0.4x_2^{t-1} + 0.6x_2^{t-2}$	2.2	3	2	$7.7 + 4.2x_6^t + 2.8x_8^t + 9.2x_9^t$	1.1
3	3	1.5	1.0	3	3	1.5	1.0
3	4	$4.3 + 5.0x_2^t + 3.0x_6^t + 0.5x_4^{t-1}$	1.7	3	4	$6.6 + 12.3x_6^t + 6.2x_8^t + 9.9x_9^t$	0.9
3	5	4.5	2.2	3	5	6.5	2.2
3	6	$0.3 + 8.0x_9^t + 0.8x_6^{t-1} + 0.19x_6^{t-2}$	1.6	3	6	$0.4 + 0.1x_6^{t-1} + 0.3x_6^{t-2} + 0.599x_6^{t-3}$	1.3
3	7	0.5	1.3	3	7	1.0	1.3
3	8	$9.8 + 5.0x_9^t + 0.9x_8^{t-1}$	0.5	3	8	$12.2 + 0.6x_8^{t-1} + 0.499x_8^{t-2}$	1.4
3	9	$2.5 + 3x_1^t$	0.8	3	9	$6.9 + 0.2x_9^{t-1} + 0.4x_9^{t-2} + 0.399x_9^{t-3}$	1.2
3	10	3.0	0.6	3	10	3.0	0.6

Table B.3: Scenario3 parameters for the synthetic data means and variances

B.2 Proofs

Since there is no change regarding the probabilistic relationships for the $\mathbf{A}$ and $\boldsymbol{\pi}$ parameters, the updating equations for these parameters stated in Appendix A.4 holds. Therefore, the focus is on determining the updating formulas od the relevancies, means and standard deviations for the relevant and irrelevant components.

Theorem B.1. Assume there is a current model $\boldsymbol{\lambda}^{(s)}$ such that the E-step has been computed with it. From optimizing Eq. (6.8), the resulting parameter $\boldsymbol{\lambda}^{(s+1)}$ can be obtained with the following updating formulas.

The feature saliencies $\{\rho_m^{(s+1)}\}_{m=1}^M$ are updated as:

$$\rho_m^{(s+1)} = \frac{\sum_{i=1}^N \sum_{t=p^*}^T \psi_m^t(i)}{T+1-p^*}. \tag{B.1}$$

The initial distribution $\boldsymbol{\pi}^{(s+1)} = \{\pi_i^{(s+1)}\}_{i=0}^N$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{B.2}$$

The transition matrix $\mathbf{A}^{(s+1)} = \{a_{ij}^{(s+1)}\}_{i,j=1}^N$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{B.3}$$

The mean and variance, $\{\epsilon_m^{(s+1)}\}_{m=1}^M$ and $\{(\tau_m^2)^{(s+1)}\}_{m=1}^M$, from the noise component, are updated as:

$$\begin{aligned} \epsilon_m^{(s+1)} &= \frac{\sum_{t=p^*}^T \sum_{i=1}^N \phi_m^t(i) x_m^t}{\sum_{t=p^*}^T \sum_{i=1}^N \phi_m^t(i)} \\ (\tau_m^2)^{(s+1)} &= \frac{\sum_{t=p^*}^T \sum_{i=1}^N \phi_m^t(i)(x_m^t - \epsilon_m)^2}{\sum_{t=p^*}^T \sum_{i=1}^N \phi_m^t(i)}. \end{aligned} \tag{B.4}$$

Setting $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ for $m = 1, ..., M$, $t = p^*, ..., T$ and hidden state $i = 1, ..., N$, the parameters $\{\eta_{imr}^{(s+1)}\}_{r=1}^{p_{im}}$ and $\{\beta_{imk}^{(s+1)}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear system:

$$\begin{cases} \sum_{t=p^*}^T \psi_m^t(i) x_m^t = \sum_{t=p^*}^T \psi_m^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^T \psi_m^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^T \psi_m^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^T \psi_m^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^T \psi_m^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^T \psi_m^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^T \psi_m^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^T \psi_m^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^T \psi_m^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{B.5}$$

if $\boldsymbol{\theta}_{im} = (\boldsymbol{\beta}_{im} | \boldsymbol{\eta}_{im})^\top$, $\boldsymbol{o}_{im}^t = (\boldsymbol{u}_{im}^t | \boldsymbol{d}_{im}^t)$, and $\boldsymbol{\Psi}_{im}^{p^*:T} := \text{Matrix}([\psi_m^{p^*}(i), ..., \psi_m^T(i)])$. Then, the previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{B.6}$$

Setting $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$; then, $\{(\sigma_{im}^2)^{(s+1)}\}_{i,m=1}^{N,M}$ can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^T \psi_m^t(i)(x_m^t - \hat{\varphi}_{im}^t)^2}{\sum_{t=p^*}^T \psi_m^t(i)}. \tag{B.7}$$

Proof. In the previous section, the proof of the updating formulas for $\mathbf{A}$ and $\boldsymbol{\pi}$ was done; therefore they are omitted here and the proof of formulas for the remaining parameters is done. In the case of $\{\rho_m\}_{m=1}^M$, compute the derivative of $\mathcal{Q}(\boldsymbol{\lambda}^{(s+1)}|\boldsymbol{\lambda}^{(s)})$ with respect to ρ_m and equate to zero:

$$\begin{aligned}
\frac{\partial Q}{\partial \rho_m} &= \sum_{t=p^*}^{T}\sum_{i=1}^{N} \frac{\psi_m^t(i)}{\rho_m} - \frac{\phi_m^t(i)}{(1-\rho_m)} = 0 \\
(1-\rho_m)\sum_{t=p^*}^{T}\sum_{i=1}^{N} \psi_m^t(i) &= \rho_m \sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i) \\
\rho_m &= \frac{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \psi_m^t(i)}{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \psi_m^t(i) + \phi_m^t(i)} \\
\rho_m^{(s+1)} &= \frac{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \psi_m^t(i)}{T+1-p^*}
\end{aligned} \tag{B.8}$$

For the mean $\{\epsilon_m^{(s+1)}\}_{m=1}^M$ from the noise component follows that:

$$\begin{aligned}
\frac{\partial \mathcal{Q}}{\partial \epsilon_m} &= \sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)\frac{(x_m^t - \epsilon_m)}{\tau_m^2} = 0 \\
\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)\epsilon_m &= \sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)x_m^t \\
\epsilon_m^{(s+1)} &= \frac{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)x_m^t}{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)}
\end{aligned} \tag{B.9}$$

Regarding the variance $\{(\tau_m^2)^{(s+1)}\}_{m=1}^M$ of the noise component:

$$\begin{aligned}
\frac{\partial \mathcal{Q}}{\partial \tau_m^2} &= \sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)\left(\frac{(x_m^t - \epsilon_m)^2}{2\tau_m^4} - \frac{1}{2\tau_m^2}\right) = 0 \\
\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)\tau_m^4 &= \sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)(x_m^t - \epsilon_m)^2\tau_m^2 \\
(\tau_m^2)^{(s+1)} &= \frac{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)(x_m^t - \epsilon_m)^2}{\sum_{t=p^*}^{T}\sum_{i=1}^{N} \phi_m^t(i)}
\end{aligned} \tag{B.10}$$

For the case of the relevant component, fix the variable X_m and the hidden state i and look for the optimum of β_{im0}:

$$\begin{aligned}
\frac{\partial \mathcal{Q}}{\partial \beta_{im0}} &= \sum_{t=p^*}^{T} \psi_m^t(i)\frac{(x_m^t - f_{im}^t)}{\sigma_{im}^2} = 0 \\
\sum_{t=p^*}^{T} \psi_m^t(i)x_m^t &= \sum_{t=p^*}^{T} \psi_m^t(i)f_{im}^t
\end{aligned} \tag{B.11}$$

For $k = 1, ..., k_{im}$, it follows that:

$$\begin{aligned} \frac{\partial \mathcal{Q}}{\partial \beta_{imk}} &= \sum_{t=p^*}^{T} \psi_m^t(i) u_{imk}^t \frac{(x_m^t - f_{im}^t)}{\sigma_{im}^2} = 0 \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{imk}^t &= \sum_{t=p^*}^{T} \psi_m^t(i) f_{im}^t u_{imk}^t \end{aligned} \tag{B.12}$$

As f_{im}^t also depends on the parameters$\{\eta_{imr}\}_{r=1}^{p_{im}}$, it holds:

$$\begin{aligned} \frac{\partial \mathcal{Q}}{\partial \eta_{imr}} &= \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-r} \frac{(x_m^t - f_{im}^t)}{\sigma_{im}^2} = 0 \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-r} &= \sum_{t=p^*}^{T} \psi_m^t(i) f_{im}^t x_m^{t-r} \end{aligned} \tag{B.13}$$

Joining the results from Eq. (B.11), Eq. (B.12) and Eq. (B.13), a linear system of equations of size $(1+k_{im}+p_{im})\times(1+k_{im}+p_{im})$ is obtained, and its solution returns the parameters $\boldsymbol{\beta}_{im}^{(s+1)}$ and $\boldsymbol{\eta}_{im}^{(s+1)}$. This must be done for $m = 1, ..., M$ and $i = 1, ..., N$. Finally, the parameters $\{\sigma_{im}^2\}_{i,m=1}^{N,M}$ are maximized as:

$$\begin{aligned} \frac{\partial \mathcal{Q}}{\partial \sigma_{im}^2} &= \sum_{t=p^*}^{T} \psi_m^t(i) \left(\frac{(x_m^t - \hat{f}_{im}^t)^2}{2\sigma_{im}^4} - \frac{1}{2\sigma_{im}^2} \right) = 0 \\ \sum_{t=p^*}^{T} \psi_m^t(i) \sigma_{im}^4 &= \sum_{t=p^*}^{T} \psi_m^t(i) (x_m^t - \hat{f}_{im}^t) \sigma_{im}^2 \\ (\sigma_{im}^2)^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \psi_m^t(i)(x_m^t - \hat{f}_{im}^t)\sigma_{im}^2}{\sum_{t=p^*}^{T} \psi_m^t(i)} \end{aligned} \tag{B.14}$$

□

Appendix C

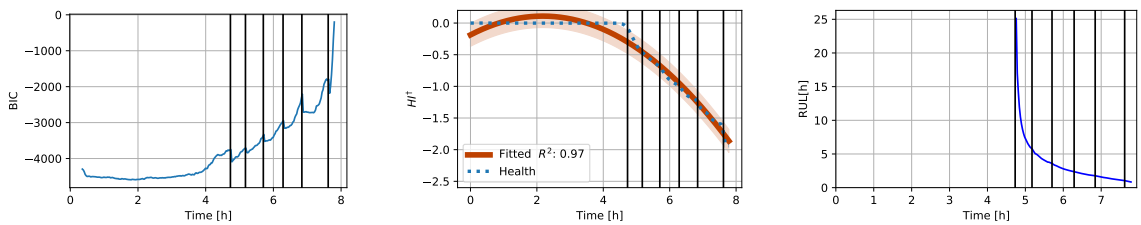
Sensitivity analysis for online ball-bearing prognosis

In this section a sensibility analysis with the hyper-parameters of the model will be performed. In the Table. C.1 different values are used for the data-stream. To be clear, 8 experiments will be performed changing each parameter and setting the others as in its reference value. These values will be used for the dataset Bearing1_1. From the experiment, the BIC, health index and RUL curves will be provided and discussed.

Parameter	Reference	Value 1	Value 2
Φ	3	2	100
p	0.1	0.5	0.9
γ_2	0.05	0.015	0.1
ζ	-2.5	-2	-3

Table C.1: Different values that are used for the sensibility analysis. In the column reference is stated the value used in the main text.

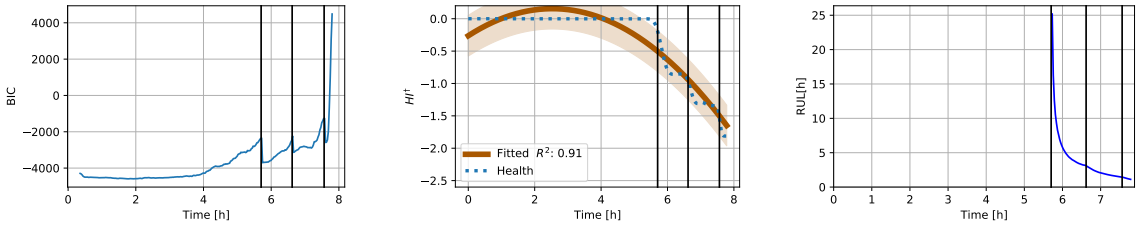
C.1 Variations of Φ



(a) (b) (c)

Figure C.1: Different result in RUL, BIC and HI when $\Phi = 2$

In Fig. C.1 it is observed the results when the parameter $\Phi = 2$. In this case, the obtained results are similar to the ones obtained when $\Phi = 3$. Nevertheless, in the case of the main



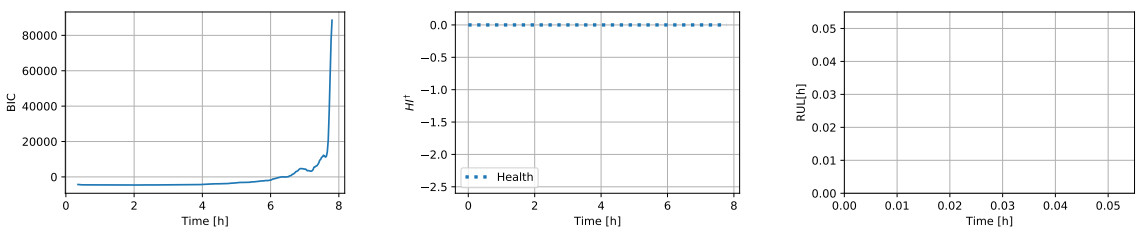
(a) (b) (c)

Figure C.2: Different result in RUL, BIC and HI when $\Phi = 10$

text, the process made 5 concept rifts, in this case, since the threshold for the test page is lower i.e. γ_1, it is easier to perform a concept drift and 6 concept drifts were performed. Additionally, in the main text, when $\Phi = 3$, the first concept drift was detected at the fifth hour, whereas in this case, the first concept drift was detected at the end of the fourth hour. In spite of these differences, the results in health index and RUL prediction were similar. Therefore, it is concluded that when $\Phi = 2$ there are redundant hidden states and additional computation steps are required to perform a model update or data inference, as consequence, $\Phi = 3$ is preferred.

In Fig. C.2 it is observed the results when $\Phi = 10$. Hence, the value of γ_1 is higher and the algorithm is lesser sensible to noise or to variations in data, this implies that it is less likely to perform or detect a concept drift. This is observed when counting the number of concept drift performed, in this case, only 3 concept drifts were performed and the health index and RUL prediction differ highly from the ones exposed in the main text. In this case, the first concept drift was detected almost at the sixth hour, the health index only got to 1.5 orders of magnitude (2 in the main text) and since the RUL began to be computed at this point, the actionable insights could be limited which is undesirable.

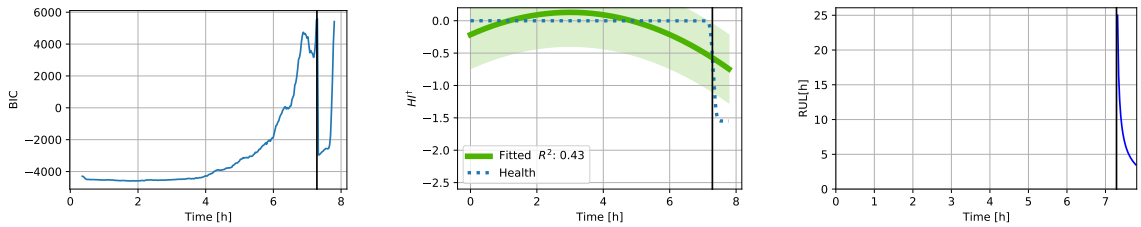
C.2 Variations of p



(a) (b) (c)

Figure C.3: Different result in RUL, BIC and HI when $p = 0.5$

Recall that p is the minimum population portion of anomalies in a sample to claim that there is a change in concept in the data in the Chernoff bounds. A change in this value will have consequences on the size of the sample, and therefore in the ability of the algorithm



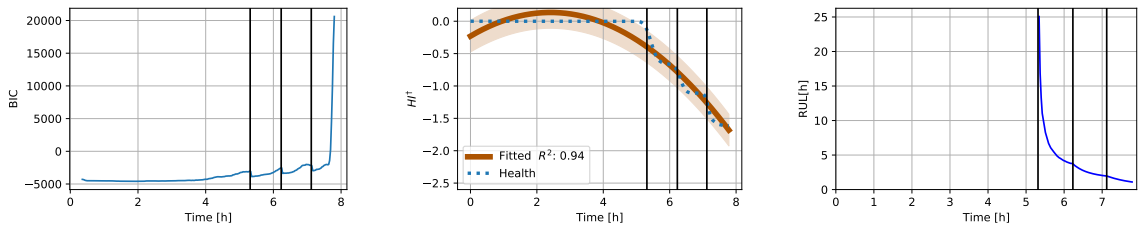
(a) (b) (c)

Figure C.4: Different result in RUL, BIC and HI when $p = 0.9$

to detect anomalies as exposed in Fig.3b of the main article. In the main text, this value is set to $p = 0.1$. In Fig. C.3 the results of the process are observed when $p = 0.5$, in this case, the sample size increases drastically and more data is required to take the decision to perform a concept drift. As consequence, as observed in (a), (b) and (c) no concept drifts were performed and no degradation was perceived. Since no degradation was detected, the health index was always constant and no RUL was computed.

In Fig. C.4 this value is set to $p = 0.9$, in this case the sample size is the same to the case when $p = 0.1$ but less to the case of $p = 0.5$ as depicted in Fig.3b of the main article. In this case, 90% of the instances in the window must be anomalies to perform a concept drift. For this scenario, only one concept drift was performed at the end of the life of the bearing and hence the RUL prediction was poor since it began to be predicted at that point. As consequence of this issues, the actionable insights were poor and this scenario is also undesirable.

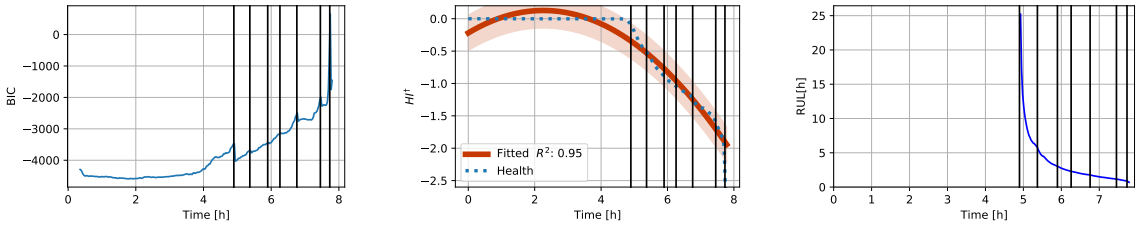
C.3 Variations of γ_2



(a) (b) (c)

Figure C.5: Different result in RUL, BIC and HI when $\gamma_2 = 0.1$

Recall that γ_2 is the level of confidence of the sample size problem for the Chernoff bounds. The smaller this value the greater the confidence of the bounds. In Fig. C.5 it is observed the results of the algorithm when $\gamma_2 = 0.1$. With this configuration, it is noticed that only 3 concept drifts were performed instead of 5 as in the main text, in spite of that, the first concept drift was performed close to the fifth operation hour as in the main article. Additionally, the health index and RUL predictions showed actionable insights similar to the



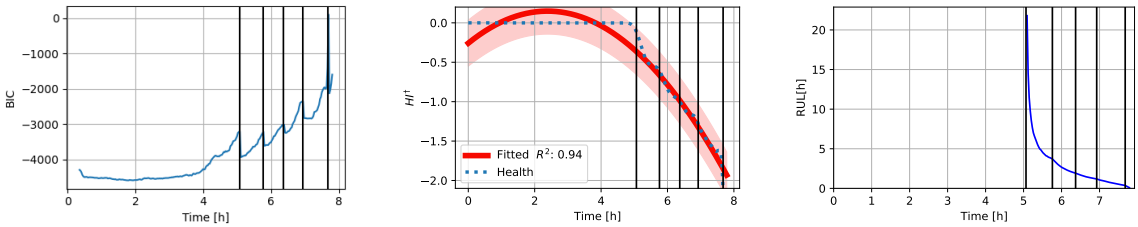
(a) (b) (c)

Figure C.6: Different result in RUL, BIC and HI when $\gamma_2 = 0.015$

ones obtained in the main text, with the exception that the health index in this configuration said that the bearing ended its life with better health.

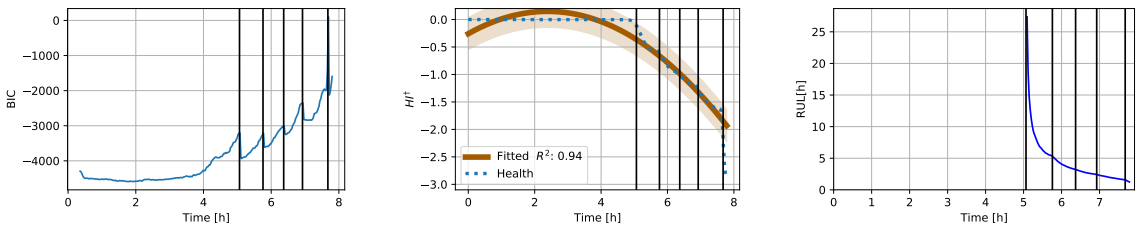
In Fig. C.6 it is pictured the results when $\gamma_2 = 0.015$. In this case the number of concept drifts increased, there were 7 concept drifts instead of 5 as in the main text. Nonetheless, the first concept drift was performed at the fifth hour as in the reference case. On the other hand, the results of health index and RUL prediction were similar to the case when $\gamma_2 = 0.05$. However, since in this case more hidden states were added to the HMM, the number of operations during the inference or model update part were increased; hence, the reference hyper-parameters configuration is preferred.

C.4 Variations of ζ



(a) (b) (c)

Figure C.7: Different result in RUL, BIC and HI when $\zeta = -2$



(a) (b) (c)

Figure C.8: Different result in RUL, BIC and HI when $\zeta = -3$

The ζ parameter is used to determine the maximum enabled orders of magnitude of degradation of a bearing. In our approach, when a ball-bearing reaches this limit it is said that it failed. In the configuration of the main article $\zeta = -2.5$, in Fig. C.7 it is shown the results when $\zeta = -2$. In this case, no changes have been observed in the concept drift detection or health index behavior, but in the RUL prediction and health index interpretation. Since the threshold is closer to zero, in this case the algorithm predicts the failure some minutes earlier to the actual failure.

In Fig. C.8 it is observed when $\zeta = -3$. in this case, more degradation is allowed before a failure is declared. In spite that the health index and novelty detection are the same as in the main text, it is observed in this case that the RUL prediction determined that the ball-bearing still had extra minutes of life when it was not true.

As a final comment, as stated in the main article, depending on the application, this threshold can be tuned depending on the ball-bearing application. It must be taken into account that ζ represents the maximum allowed orders of magnitude of difference of the used features compared to a normal or custom behavior.

Appendix D

Supplementary material for online LFS-HMMs

D.1 Proofs

Theorem D.1. Assume there is a current model $\boldsymbol{\lambda}^{(s)}$ from which the E-step has been computed in the formulas Eq. (9.9). By maximizing Eq. (9.8), the resulting parameter $\boldsymbol{\lambda}^{(s+1)}$ can be obtained with the following updating formulas.

The feature saliencies $\{\rho_{im}^{(s+1)}\}_{m=1}^{M}$ are updated as:

$$\rho_{im}^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_m^t(i)}{\sum_{t=p^*}^{T} \gamma^t(i)}. \tag{D.1}$$

The initial distribution $\boldsymbol{\pi}^{(s+1)} = \{\pi_i^{(s+1)}\}_{i=0}^{N}$ is updated as:

$$\pi_i^{(s+1)} = \gamma^{p^*}(i). \tag{D.2}$$

The transition matrix $\boldsymbol{A}^{(s+1)} = \{a_{ij}^{(s+1)}\}_{i,j=1}^{N}$ is updated as:

$$a_{ij}^{(s+1)} = \frac{\sum_{t=p^*}^{T-1} \xi^t(i,j)}{\sum_{t=p^*}^{T-1} \gamma^t(i)}. \tag{D.3}$$

The mean and variance, $\{\epsilon_m^{(s+1)}\}_{m=1}^{M}$ and $\{(\tau_m^2)^{(s+1)}\}_{m=1}^{M}$, from the noise component, are updated as:

$$\begin{aligned} \epsilon_m^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i) x_m^t}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)} \\ (\tau_m^2)^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)(x_m^t - \epsilon_m)^2}{\sum_{t=p^*}^{T} \sum_{i=1}^{N} \phi_m^t(i)}. \end{aligned} \tag{D.4}$$

Denoting $\varphi_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}$ for $m = 1, ..., M$, $t = p^*, ..., T$ and hidden state $i = 1, ..., N$,

the relevance parameters $\boldsymbol{\eta}_{im} = \{\eta_{imr}\}_{r=1}^{p_{im}}$, $\boldsymbol{\beta}_{im} = \{\beta_{imk}\}_{k=0}^{k_{im}}$ can be updated jointly, solving the following linear equation system:

$$\begin{cases} \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t = \sum_{t=p^*}^{T} \psi_m^t(i) \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{im1}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{im1}^t \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t u_{imk_{im}}^t = \sum_{t=p^*}^{T} \psi_m^t(i) u_{imk_{im}}^t \varphi_{im}^t \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-1} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-1} \varphi_{im}^t \\ \quad\vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \\ \sum_{t=p^*}^{T} \psi_m^t(i) x_m^t x_m^{t-p_{im}} = \sum_{t=p^*}^{T} \psi_m^t(i) x_m^{t-p_{im}} \varphi_{im}^t \end{cases} \tag{D.5}$$

if $\boldsymbol{\theta}_{im} = [\boldsymbol{\beta}_{im} | \boldsymbol{\eta}_{im}]^\top$, $\boldsymbol{o}_{im}^t = [\boldsymbol{u}_{im}^t | \boldsymbol{d}_{im}^t]$, and $\boldsymbol{\Psi}_{im}^{p^*:T} := \text{Matrix}([\psi_m^{p^*}(i), ..., \psi_m^T(i)])$. Then, the previous linear system is solved as:

$$\boldsymbol{\theta}_{im}^{(s+1)} = \left((\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{o}_{im}^{p^*:T} \right)^{-1} (\boldsymbol{o}_{im}^{p^*:T})^\top \boldsymbol{\Psi}_{im}^{p^*:T} \boldsymbol{x}_m^{p^*:T} \tag{D.6}$$

Setting $\hat{\varphi}_{im}^t := \boldsymbol{u}_{im}^t \boldsymbol{\beta}_{im}^{(s+1)} + \boldsymbol{d}_{im}^t \boldsymbol{\eta}_{im}^{(s+1)}$; then, $\{(\sigma_{im}^2)^{(s+1)}\}_{i,m=1}^{N,M}$ can be updated as:

$$(\sigma_{im}^2)^{(s+1)} = \frac{\sum_{t=p^*}^{T} \psi_m^t(i)(x_m^t - \hat{\nu}_{im}^t)^2}{\sum_{t=p^*}^{T} \psi^t(i)}. \tag{D.7}$$

Proof. Deriving $\mathcal{Q}_2(\boldsymbol{\lambda}^{(s+1)} | \boldsymbol{\lambda}^{(s)})$ with respect to ρ_{im}:

$$\begin{aligned} \frac{\partial \mathcal{Q}_2}{\partial \rho_{im}} &= \sum_{t=p^*}^{T} \frac{\psi_m^t(i)}{\rho_{im}} - \frac{\phi_m^t(i)}{(1-\rho_{im})} = 0 \\ (1-\rho_{im}) \sum_{t=p^*}^{T} \psi_m^t(i) &= \rho_{im} \sum_{t=p^*}^{T} \phi_m^t(i) \\ \rho_{im} \left(\sum_{t=p^*}^{T} \psi_m^t(i) + \phi_m^t(i) \right) &= \sum_{t=p^*}^{T} \psi_m^t(i) \\ \rho_{im} &= \frac{\sum_{t=p^*}^{T} \psi_m^t(i)}{\sum_{t=p^*}^{T} \psi_m^t(i) + \phi_m^t(i)} \\ \rho_{im}^{(s+1)} &= \frac{\sum_{t=p^*}^{T} \psi_m^t(i)}{\sum_{t=p^*}^{T} \gamma^t(i)} \end{aligned} \tag{D.8}$$

□

D.2 Synthetic data parameters

The parameters of mean and variance are exposed in Table D.1. Note that, the dependencies of some variables change for every hidden states. Meanwhile, the parameters of the noise variables are the same for all the hidden states.

i	m	φ_{im}	σ_{im}	i	m	φ_{im}	σ_{im}
1	1	1.0	0.2	2	1	3.0	0.9
1	2	1.0	0.3	2	2	$4.0 + 0.2X_1^t$	0.8
1	3	1.0	1.0	2	3	1.0	1.0
1	4	1.0	0.6	2	4	2.0	0.7
1	5	1.0	0.2	2	5	$2.0 + 0.4X_4^t$	0.5
1	6	1.0	1.0	2	6	1.0	1.0
1	7	1.0	0.4	2	7	2.0	0.6
1	8	1.0	0.5	2	8	$3.0 + 0.4X_7^t$	0.7
1	9	1.0	1.0	2	9	1.0	1.0
1	10	1.0	1.0	2	10	2.0	1.1
3	1	3.0	1.0	4	1	$2.0 + 0.5X_1^{t-1}$	1.2
3	2	$2.0 + 0.3X_1^t + 0.4X_4^t + 0.5X_2^{t-1}$	0.9	4	2	$3.0 + 0.2X_1^t + 0.3X_4^t + 0.3X_2^{t-1}$	1.0
3	3	1.0	1.0	4	3	4.0	1.2
3	4	$2.3 + 0.6X_4^{t-1}$	0.8	4	4	$2.5 + 0.3X_{10}^t + 0.5X_4^{t-1}$	0.9
3	5	$2.5 + 0.4X_4^t + 0.3X_7^t$	0.5	4	5	$3.0 + 0.4X_4^t + 0.3X_7^t$	0.7
3	6	1.0	1.0	4	6	1.0	1.0
3	7	3.0	0.8	4	7	$2.6 + 0.5X_{10}^t$	0.8
3	8	$3.5 + 0.4X_7^t$	0.8	4	8	$3.0 + 0.4X_7^t + 0.5X_8^{t-1}$	0.9
3	9	1.0	1.0	4	9	1.0	1.0
3	10	$3.0 + 0.7X_{10}^{t-1}$	0.9	4	10	6.0	0.7
5	1	$2.0 + 0.5X_1^{t-1} + 0.2X_1^{t-2}$	0.6				
5	2	$1.0 + 0.2X_1^t + 0.3X_4^t + 0.4X_2^{t-1} + 0.1X_2^{t-2}$	0.6				
5	3	6.0	1.5				
5	4	$2.8 + 0.3X_{10}^t + 0.6X_4^{t-1} + 0.2X_4^{t-2}$	0.7				
5	5	$3.5 + 0.1X_2^t + 0.3X_4^t + 0.2X_7^t + 0.3X_5^{t-1}$	0.9				
5	6	1.0	1.0				
5	7	$2.8 + 0.3X_{10}^t + 0.2X_7^{t-1}$	0.5				
5	8	$3.0 + 0.1X_5^t + 0.2X_7^t + 0.3X_8^{t-1} + 0.1X_8^{t-2}$	0.4				
5	9	1.0	1.0				
5	10	$6.0 + 0.6X_{10}^{t-1}$	0.6				

Table D.1: Used parameters for synthetic data

Appendix E

Supplementary material for KDE-AsHMMs

E.1 Parameters to generate synthetic data

Here, the parameters used to generate the synthetic data are exposed, isntead of describing the values $\{\{c_{imk}\}_{k=1}^{\kappa_{im}}, \{d_{imr}\}_{r=1}^{p_{im}}, e_{im}, \sigma_{im}\}_{i=1,m=1}^{N,M}$, the mean formulas and variances for each variable and for each hidden state are provided.

Par	Value	Par	Value	Par	Value
$\mu_{1,0}^t$	0.0	$\mu_{2,0}^t$	2.0	$\mu_{3,0}^t$	$-2.0 + 0.4x_0^{t-1}$
$\mu_{1,1}^t$	-10.0	$\mu_{2,1}^t$	$-1.0 + 1.5(x_0^t - 2.0)^2 + 0.5x_1^{t-1}$	$\mu_{3,1}^t$	$2.0 - 0.9(x_0^t + 2.0)^2 + 0.4x_1^{t-1} + 0.4x_1^{t-2}$
$\mu_{1,2}^t$	20.0	$\mu_{2,2}^t$	$3.0 - 0.9(x_0^t - 2.0)^2$	$\mu_{3,2}^t$	$-3.0 + 1.5(x_0^t + 2.0)^2 + 0.4x_2^{t-1}$
$\mu_{1,3}^t$	0.0	$\mu_{2,3}^t$	$2.0 + 0.6x_3^{t-1}$	$\mu_{3,3}^t$	$-2.0 - 0.5x_3^{t-1} - 0.3x_3^{t-2}$
$\mu_{1,4}^t$	8.0	$\mu_{2,4}^t$	$2.0 + 2.0(x_3^t - 2.0)^2$	$\mu_{3,4}^t$	$-2.0 - 2.0(x_3^t + 2.0)^2 + 0.6x_4^{t-1}$
$\mu_{1,5}^t$	1.0	$\mu_{2,5}^t$	1.0	$\mu_{3,5}^t$	1.0
$\mu_{1,6}^t$	2.0	$\mu_{2,6}^t$	2.0	$\mu_{3,6}^t$	2.0
$\sigma_{1,0}$	0.1	$\sigma_{2,0}$	1.0	$\sigma_{3,0}$	2.0
$\sigma_{1,1}$	0.3	$\sigma_{2,1}$	0.7	$\sigma_{3,1}$	0.5
$\sigma_{1,2}$	0.5	$\sigma_{2,2}$	0.2	$\sigma_{3,2}$	0.6
$\sigma_{1,3}$	0.2	$\sigma_{2,3}$	1.2	$\sigma_{3,3}$	1.3
$\sigma_{1,4}$	1.8	$\sigma_{2,4}$	1.4	$\sigma_{3,4}$	0.2
$\sigma_{1,5}$	0.5	$\sigma_{2,5}$	0.5	$\sigma_{3,5}$	0.5
$\sigma_{1,6}$	0.6	$\sigma_{2,6}$	0.6	$\sigma_{3,6}$	0.6

Table E.1: Synthetic parameters to generate KDE-AsHMM data