Dynamic Bayesian Networks: Estimation, Inference and Applications

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Igor Melnyk

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Arindam Banerjee

June, 2016

Acknowledgements

I would like to thank my advisor, Arindam Banerjee, for supporting and guiding me through the years of my PhD. He has been an inspiration to me, sharing with me his wisdom and experience, providing ideas and motivation, encouraging during the difficult times and sharing the joy after accomplishing the goals. I have learned a great deal of machine learning from him and benefited a lot from his skill and intuition at modeling difficult problems. I am grateful to him for providing freedom to explore many research endeavors, which resulted in successful algorithms and papers.

I would also like to thank my dissertation committee members, Vipin Kumar, Yousef Saad and Mihailo Jovanovic for their valuable feedback and ideas during the preparation of this thesis. I am also grateful to my collaborators at NASA, Bryan Matthews, Nikunj Oza and Hamed Valizadegan. It was a pleasure to work and collaborate with them on the anomaly detection problems throughout almost four years of the project.

On the personal side, I would like to thank all my labmates and friends at the University for good company and interesting discussions over the years including Huahua Wang, Andre Goncalves, Soumyadeep Chatterjee, Vidyashankar Sivakumar, Farideh Fazayeli, Qiang Fu, Nisheeth Srivastava, Nicholas Johnson, Amir Taheri, Karthik Subbian, Puja Das, Sheng Chen, Konstantina Christakopoulou, Shaozhe Tao, Qilong Gu and many others. I am grateful to know you. I loved our discussions about research, politics, culture, sports and many other topics.

Finally, I would like to thank my family for providing support and encouragements over the many years while pursuing my research. I am incredibly grateful to my wife Zina for love and support and for understanding all the late nights and weekends that went into finishing my PhD.

Dedication

To my parents and wife.

Abstract

In recent years, there has been a significant increase in the applications dealing with dynamic, high-dimensional, heterogeneous data streams. For example, in the domains such as healthcare, activity recognition, aviation systems, etc. multiple sensors provide a record of many continuous and discrete parameters over long periods of time, and the objective is to monitor behavior of the objects, discover meaningful patterns or detect anomalous events.

In spite of a vast literature on data mining and machine learning techniques, these problems have continued to remain difficult. Primarily this is due to a challenge of proper characterization of the interdependencies between multiple data sources, being a mixture of continuous and discrete type. Moreover, for applications that deal with data monitoring or unusual behavior detection, the additional challenge is a design of discovery algorithms aimed at extracting patterns, trends, anomalies in unsupervised settings where data is commonly noisy and even partially unobservable.

In this work, we propose a suite of models and methods for the analysis of such data by using a Dynamic Bayesian Network (DBN) representation. DBN is a general tool for establishing dependencies between variables evolving in time, and is used to represent complex stochastic processes to study their properties or make predictions on the future behavior. The main challenge in using DBN is to identify a model structure, learn its parameters with estimation guarantees and perform efficient inference. Our work has made advances in addressing the above problems, especially in the context of anomaly detection, by proposing several frameworks for anomaly detection in multivariate time series data and building efficient algorithms for learning and inference.

In the first part of the thesis, we present a framework for modeling dynamic discrete sequences based on a hidden semi-Markov model (HSMM). We chose HSMM due to its inherent ability to model durations in addition to latent state transitions based on the observed sequences, where such modeling is frequently used in the areas such as activity recognition or object monitoring. An important aspect of using HSMM is the parameter learning and inference. For this purpose, we introduce a novel spectral algorithm to perform inference in HSMM. Unlike expectation maximization (EM), our approach correctly estimates the probability of a given observation sequence based on a set of training sequences. Moreover, the algorithm provides estimation guarantees and is computationally efficient.

In the second part, we consider modeling and anomaly detection in the continuous multivariate time series data. For this purpose, we present a framework where each data object is represented using a vector autoregressive (VAR) model. A similarity neighborhood graph is then constructed based on the constructed VAR models and anomaly detection is then applied to identify abnormal events. A key step in the above framework is the estimation of the parameters in the VAR model, usually formulated as a least-squares optimization problem with a regularization based on the norms such as Lasso, group Lasso, order weighted Lasso, etc. We study the properties of such optimization problem and establish bounds on the non-asymptotic estimation error of the VAR parameters.

Finally, in the last part, we combine the ideas of the semi-Markov modeling of discrete sequences and autoregressive modeling of continuous data, and represent the multivariate heterogeneous time series data of the flight using semi-Markov switching vector autoregressive (SMS-VAR) model. Detection of anomalies is then based on measuring dissimilarities between the model's prediction and data observation.

The evaluation of the proposed frameworks is done on the NASA flight dataset, containing over a million of flights, representing 35 different aircrafts. For each flight, the data has a record of over 300 parameters, sampled at 1 Hz, including sensor readings, control inputs and weather information. The objective is then to detect anomalous flight segments due to mechanical, environmental, or human factors. Extensive experimental results on this dataset illustrate that the proposed frameworks can detect various types of anomalies along with the key parameters involved and outperforms the current state-of-the-art approaches.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Given data about some process or phenomenon, such as speech, a protein sequence, or a stock market, one might be interested in constructing a representative model in order to study its properties or to make predictions about its future behavior. One of the most general models one could build is to construct the underlying probability distribution which generated the data. Such distribution would establish the relationship between various parameters of the phenomenon as well as govern its evolution in time. For the ease of use and to visualize and study such relationships, Bayesian Networks (BNs) [6] are usually constructed, which are simply a graphical way to represent the static dependencies between the variables. To characterize a temporal component of the process, the dynamics is added to BNs and these models are then called Dynamic Bayesian Networks (DBNs) [7]. For example, consider Figure 1.1, which illustrates a DBN of some abstract process evolving across three time steps. Within each time stamp, the circles denote the variables representing the model and the black arrows show the static relationship among them. The blue arrows, on the other hand, represent the dynamics and show how the system evolves in time.

The main objective of this work is the DBNs for multivariate, heterogeneous time series data. We are interested in the models that can efficiently represent the dynamics of the system with multiple, interdependent parameters evolving over a long period of time. Such data frequently arises in aviation [8], industrial [9], medical [10] or economic [11] domains, characterized by high dimensional, high throughput systems recording large amounts of data in a short period of time. In modeling such data, we focus

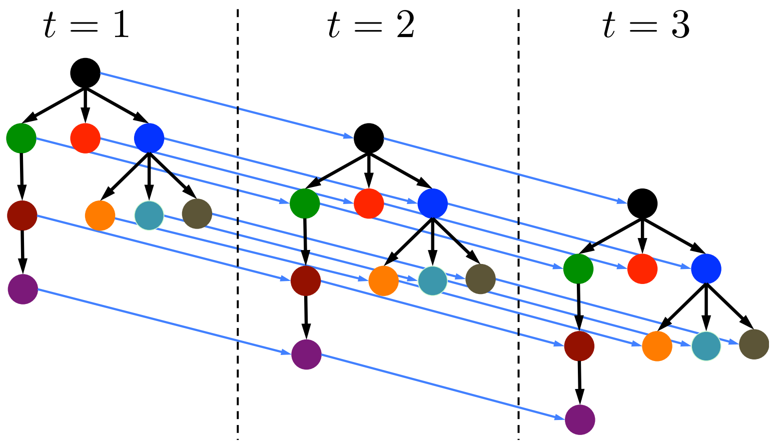


Figure 1.1: Dynamic Bayesian Network presentation of an abstract process evolving across three time steps. The circles represent the variables of the model and the arrows show dependencies between the variables.

specifically on the models enabling accurate short- and long-term forecasting as well as the models which are useful for analyzing the properties of the underlying system and studying the complex relationships in the data. Moreover, we are interested in the efficient algorithms to estimate the DBN models as well as to perform fast online computations of the probabilistic queries in the constructed models. Given the large amounts of data generated by the systems, often contaminated by noise, the algorithms need to be robust, scalable, computationally efficient and with estimation guarantees.

1.1 Existing Approaches

There are three key problems associated with using DBNs: (i) structure learning, which focuses on finding the graph structure (e.g., the tree-like structure within each time stamp in Figure 1.1) that encodes the conditional dependence and indepedence in the data; (ii) parameter learning, which computes the parameters of the probability distributions specified by DBN and (iii) inference, consists of answering various queries about the underlying process, which are usually marginal or posterior probabilities of the variables of interest.

The structure learning is usually addressed using three main approaches. The constraint-based structure learning approach [12, 13] tries to test for conditional dependence and independence in the data and then find a network that best explains these dependencies and independencies. This method is known to be sensitive to failures in individual independence tests, for example if one of the tests returns a wrong answer then the constructed network no longer represents the data it models. The second approach is a score-based structure learning [14, 15], which addresses the learning as a model selection problem. In particular, a set of possible network structures is first identified and then a scoring function measuring how well the model fits the data is applied to select the best fitting model. The disadvantage of the score-based approach is that it poses a search problem that may have to search a very large space of structures, making it computationally infeasible. Finally, the hybrid approach [16, 17] uses both the conditional independence tests to reduce the space of candidate structures and scores to identify the optimal structure among them.

Once the DBN structure is determined, the next step is to estimate the parameters of the distribution specifying the model. There are usually two main approaches for this: one based on Maximum Likelihood (ML) or Maximum a Posteriori (MAP) estimation, which do a point estimate, and the other using a full Bayesian approach. In the ML approach [6] it is assumed that we have access to the training data $\mathcal{D}$ and that the model is specified in terms of a set of parameters Θ defining the underlying probability distribution. The likelihood $p(\mathcal{D}|\Theta)$ is constructed and then $\hat{\Theta}$ is found which maximizes the likelihood over the parameter space

$$\hat{\Theta}_{ml} = \arg\max_{\Theta} p(\mathcal{D}|\Theta).$$

If a prior knowledge about the model parameters is known, then a MAP approach can be employed, which aims at finding the parameter $\hat{\Theta}_{map}$, which maximizes the posterior $p(\Theta|\mathcal{D})$

$$\hat{\Theta}_{map} = \arg\max_{\Theta} p(\Theta|\mathcal{D}) = \arg\max_{\Theta} \frac{p(\mathcal{D}|\Theta)p(\Theta)}{p(\mathcal{D})} = \arg\max_{\Theta} p(\mathcal{D}|\Theta)p(\Theta),$$

where $p(\Theta)$ is the prior distribution on the model parameters reflecting our prior knowledge on Θ. Also observe that in the last equality we dropped the term $p(\mathcal{D})$ since it has no functional dependence on Θ.

Both ML and MAP estimations return only a single and specific value of the parameter, Θ. In contrast, in full Bayesian estimation the objective is to compute the posterior distribution

$$p(\Theta|\mathcal{D}) = \frac{p(\mathcal{D}|\Theta)p(\Theta)}{p(\mathcal{D})}.$$

The advantage of the Bayesian approach over the point estimators is that we get a probability distribution $p(\Theta|\mathcal{D})$, which completely specifies all the possible model parameters and it is up to us to decide which one to select (e.g., we may choose the expected value of this distribution if the variance is small enough). On the other hand, the Bayesian approach is computationally infeasible in many cases since it requires the computation of $p(\mathcal{D}) = \int_\Theta p(\mathcal{D}|\Theta)p(\Theta)d\Theta$, an intractable integration for many practical scenarios [6].

In situations when the given training data $\mathcal{D}$ does not have a record of all the relevant parameters and some of the variables might be hidden from the observer, a standard approach is to use the Expectation Maximization (EM) [18, 19] algorithm. It is an iterative approach whose main functionality can be described as follows. Denote by $\mathcal{D}$ the data about the observed variables and by Z the set of hidden variables in the DBN model. If we did have access to the data about these variables, then our objective would be to determine the parameters which maximize the (log) likelihood $\log p(\mathcal{D}, Z|\Theta)$ of the data. However, since we do not have access to Z directly, what we can do is to maximize the expectation of $\log p(\mathcal{D}, Z|\Theta)$

$$\mathbb{E}_Z[\log p(\mathcal{D}, Z|\Theta)] = \sum_Z \log p(\mathcal{D}, Z|\Theta)p(Z|\mathcal{D}, \hat{\Theta}) = Q(\Theta, \hat{\Theta}),$$

where $\hat{\Theta}$ is some initial estimate of the parameters. Note that the resulting expectation can be viewed as a function $Q(\Theta, \hat{\Theta})$, parameterized by Θ. Now, similarly as in ML approach, we maximize the expectation of the likelihood

$$\hat{\hat{\Theta}} = \arg\max_\Theta Q(\Theta, \hat{\Theta}),$$

where $\hat{\hat{\Theta}}$ is now a new parameter estimate. This procedure is then repeated until convergence, i.e., given an initial estimate of the parameters at iteration k, $\hat{\Theta}_k$, we repeat the following two steps (expectation (E) and maximization (M)) until convergence

- E step: given $\hat{\Theta}_k$ and $\mathcal{D}$, compute $p(Z|\mathcal{D}, \hat{\Theta}_k)$;

- M step: compute $Q(\Theta, \hat{\Theta}_k)$ and get $\hat{\Theta}_{k+1} = \arg\max_{\Theta} Q(\Theta, \hat{\Theta}_k)$.

Finally, the third important problem associated with DBNs is inference, i.e., computing the probabilistic queries of the variables of interest. The approaches for this problem usually fall into two main categories: the exact inference and approximate inference. For the exact inference, the Junction Tree algorithm [7] is commonly used, which helps to decompose the global computations of joint probability into a linked set of local computations by converting a DBN into a tree-structured graph. In this approach, the DBN is first converted into a Junction tree (also called Clique tree) [20] and then a sum-product algorithm [21] is applied to compute the quantities of interest. The exact inference approach suffers from high computational and space requirements since their complexity is exponential in the tree-width [6] of the network. For this reason, for the networks with large tree-width, approximate inference approaches are usually applied.

The approximate inference approaches can be classified into stochastic and deterministic approximations. The stochastic approximation methods are based on numerical sampling and are generally known as Monte Carlo techniques. The main idea is to approximate the intractable probability distribution with samples, e.g., approximating the population mean with an average of the data points drawn from the corresponding probability distribution. A popular class of sampling algorithms are based on Markov Chain Monte Carlo (MCMC) methods, which perform sampling from a probability distribution based on constructing a Markov chain that has a desired distribution as its steady-state. Examples of MCMC approaches are the Metropolis-Hastings algorithm [22, 23], Gibbs sampling [24], slice sampling [25], etc. The deterministic approaches, based on variational inference techniques [19, 26], in contrast to the sampling approaches, are based on approximating the target probability distribution with another distribution analytically. The main idea is to first pick a family of approximating distribution, which is parameterized by certain parameters. Then, the parameters are varied such that the approximation is close to the target, which is then used as a proxy to make the probabilistic queries of interest.

Unfortunately, for an arbitrary process or phenomenon, identifying the structure of the network, learning its model parameters with estimation guarantees and performing efficient inference are challenging and in many cases intractable problems [19]. For example, the estimation of the optimal DBN structure for a complicated process with limited and noisy data is usually an intractable problem [6]. Moreover, the data might not have a record of all the relevant parameters and some of the variables might be unobserved. In this case, DBNs with latent variables are employed, for which the structure identification becomes even a more non-trivial task. The parameter estimation problem is again a challenging task, especially when a part of the data related to latent variables is missing. The approaches based on EM [18] are the most widely employed for these scenarios, however, these are iterative approaches crucially relying on initialization with limited convergence or estimation guarantees. Even when we have access to all the data about a process, some existing techniques for computing the model parameters still lack theoretical guarantees for the obtained estimates. For example, when modeling time series data, the parameter estimator is usually formulated as a linear regression on correlated and dependent data [27] and the statistical analysis of the solution of a general regularized regression is still missing. Finally, the exact inference problem for arbitrary DBNs is computationally intractable [7]. Approximate inference approaches based on sampling or variational methods have been proposed to alleviate the problem [28], however, as we discussed above, these techniques are based on approximating probability distribution and can still be computationally prohibitive.

1.2 Our Work

In this work we attempt to advance the current state of the art in closing the gap to the goal of being able to model a probability distribution with provable guarantees for structure identification, parameter learning and inference. In particular, we have proposed a spectral algorithm for learning and inference in hidden semi-Markov model (HSMM) [29]. This algorithm, in contrast to EM, is non-iterative, computationally efficient and has provable estimation guarantees. We have also studied the problem of estimation of structured vector auto-regressive (VAR) models for time series data where the structure can be captured by any suitable norm, e.g., Lasso [30], group Lasso [31], order weighted Lasso [32], sparse group Lasso, etc. For this type of models we have proved [33] non-asymptotic error bounds and established the relationship between accuracy of estimated parameters and the number of required samples (size of the training data).

The main application area of our work is the anomaly detection in aviation systems and the specific process that we study is the flight of aircraft. In this context, the goal is to detect anomalous flight segments, due to mechanical, environmental, or human factors in order to identifying operationally significant events and provide insights into the flight operations and highlight otherwise unavailable potential safety risks and precursors to accidents. For this application, we have proposed a number of approaches for detection of safety events based on HSMM [34, 29], VAR [35] and switching VAR models [36].

The primary motivation for our choice of this application area is the realization of the importance of the aviation safety. In particular, it is estimated that by 2040 the United States alone can expect an increase of more than 60% in the commercial air traffic [37]. The anticipated air traffic growth can lead to increased congestions on the ground and in the air, creating conditions for possible incidents or accidents. Noting this problem, air transportation authorities are engaged in research and development of the Next Generation Air Transportation System [38, 39], the initiative to improve air traffic control system by increasing its capacity and utilization. A part of this effort is devoted to the processing and analysis of the air traffic flight information, also known as Flight Operations Quality Assurance (FOQA) data, to detect issues in aircraft operation, study pilot-automation interaction problems, propose corrective actions or design new training procedures.

The currently deployed automated methods for the analysis of FOQA data are usually exceedance-based approaches [40, 41], which monitor the normal operation of the flight using predefined ranges on the parameters and any deviations outside of these ranges are flagged as anomalies. Although this approach is simple and fast, it is limited since the method examines each feature independently of the others, ignoring potential correlations among the parameters. Moreover, since the thresholds need to be defined upfront, this method can fail to discover previously unknown abnormal events.

The main source of data in our work is the real NASA flight dataset [42]. It contains over a million of flights of 35 aircrafts from a partner airline company. For each flight, the data has a record of 186 parameters, sampled at 1 Hz, including sensor readings control inputs and weather information. A diagram in Figure 1.2 shows some of the flight data parameters and the relationships among them. They key characteristics of these data can be summarized as follows

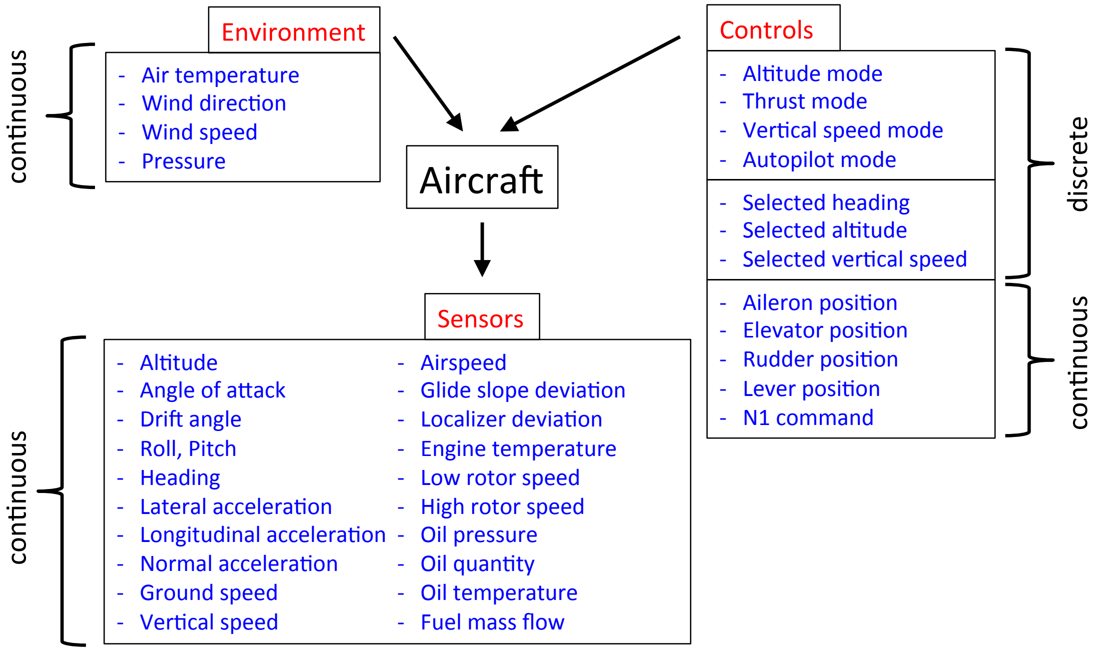


Figure 1.2: Diagram showing flight data parameters and the relationships among them. The inputs (environment and control parameters) determine the behavior of the aircraft and their effects are registered by the sensors, which are the outputs of the system.

- Multivariate
- Variable length
- Heterogeneous
- Unlabeled.

Specifically, each flight is represented as a multivariate time series. Since each of them has a different duration, the length of multivariate time series varies. Moreover, some of the flight parameters are continuous (e.g., some environment and sensor measurements) while others are discrete (e.g., some of the control parameters), therefore the data is heterogeneous. Finally, the dataset has no ground truth information, i.e., it is not known which of the flights are normal and which are anomalous. Consequently, given the above description it is clear that the task of anomaly detection in such settings is challenging.

We approach the above problem by breaking it into subproblems, solving them and then combine the results to get a solution to the original problem. First, in Chapter 2 we present a method for anomaly detection in discrete sequences based on HSMM. The key problems which also need to be addressed in using HSMM are parameter learning and inference. For this purpose, in Chapter 2 we also present a spectral algorithm for inference in these type of models. Next, in Chapter 3 we consider the same problem using only the continuous features and propose a vector autoregressive model-based anomaly detection framework. The key step in this framework is the estimation of the VAR model, which is usually done by solving a regularized least-squares optimization problem. In Chapter 4 we present a theoretical analysis studying the properties of this optimization problem from the statistical point of views, i.e., under which conditions on the data the VAR estimation problem is guaranteed to produce the accurate estimates. Finally, in Chapter 5, we propose the anomaly detection method for heterogeneous flight data, which is based on semi-Markov switching vector autoregressive (SMS-VAR) modeling and relies on the earlier developed ideas of HSMM and VAR.

We note that our proposed anomaly detection techniques are not limited to aviation safety domains and can be also used in other areas such as network intrusion detection [43], fraud detection [44], public and healthcare domain anomaly detection [45], etc. Moreover, the HSMM, for which we designed efficient spectral algorithm for learning and inference, is a popular modeling framework in many areas, including activity recognition [46], speech synthesis [47], modeling web browsing behavior [48], etc. The VAR model, whose structured estimation was analized in this work, is a widely used modeling framework, whose applications range from describing the behavior of economic and financial time series [49] to modeling the dynamical systems in the control theory [50] to estimating brain function connectivity [51] and many others. Finally, the switching VAR model merges ideas from HSMM and VAR to enable modeling of systems whose dynamics can transition in a discrete manner from one linear operating regime to another. Such ideas found applications in economics [52], healthcare [53], signal processing [54], etc.

Chapter 2

Hidden semi-Markov Model: Discrete Data Modeling

In this chapter we present the work which addresses the problem of modeling dynamic discrete flight data and in Section 2.2 we show how hidden semi-Markov model (HSMM) can be used to perform anomaly detection in such data. In Sections 2.3 and 2.4 we derive efficient non-iterative spectral algorithm for inference in HSMM. We present experimental results in Section 3.4, comparing the proposed spectral algorithm with EM on synthetic and real flight data.

2.1 Introduction

The discrete data usually correspond to the pilot actions which control the behavior of the aircraft. Pilot actions have certain unique aspects which make the modeling typical behavior as well as detecting anomalies a challenge. While the actions are chosen from a fixed alphabet of possible actions, the typical/normal actions are neither totally ordered nor memoryless. The actions can be considered weakly ordered in a specific phase of a flight, and certain set of actions are common in certain phases, such as take-off, cruise, and landing. Further, the duration between subsequent actions may vary, and there may not be any action in every discrete time-step, e.g., every second. To capture such weakly ordered actions with variable durations we proposed to use hidden semi-Markov models (HSMMs) [55], [56].

HSMM is an extension of a simpler hidden Markov model (HMM) [57], whose main drawback as a model for pilot actions is that it encourages fast hidden state switches. Subsequent pilot actions usually have a time interval, and such intervals constitute normal behavior. Therefore, it is difficult for HMM to model such intervals as being in the same latent state for prolonged periods of time. HSMM, on the other hand addresses this problem by introducing additional hidden variable which controls duration of the hidden state. With this modifications, the HSMM allows arbitrary distributions of state durations, thus improving modeling of pilot actions.

The problem of detecting anomalies in the pilot actions has attracted the attention of many researchers. For example, Budalakoti et al. [58] addressed the problem of anomaly detection by clustering the training action sequences using k-medoids algorithm into groups based on the normalized longest common subsequence (LCS) similarity measure. The anomaly score was based on the similarity measure of a test sequence to the closest cluster medoid.

Anomaly detection algorithm based on Dynamic Bayesian Networks was proposed by Saada et al. [59], where hidden variables correspond to pilot actions and observable variables model the instruments data. Unfortunately, the authors proposed to train their model using sequences which contained information about manually annotated hidden nodes, which is impractical for many realistic scenarios.

Srivastava [60] addressed the problem of anomaly detection in pilots actions by using Hidden Markov Models. The observations are modeled using N-dimensional binary vector corresponding to binary switches in an aircraft cockpit. Since the number of possible switches could be large, to learn the model, they first perform clustering of the training data to get a smaller class of discrete observations and then apply HMM over the reduced data. However, as we already mentioned, HMM has a restriction to geometric distribution of persistence in the same hidden state, thus limiting its ability to properly model the pilots actions.

2.2 Description of Anomaly Detection Approach

The presented approach uses normal sequences to learn a model that captures normal behavior. For any test sequence, the likelihood of the sequence to have come from the model is used to determine if the sequence is anomalous or not. In particular, high likelihood sequences are deemed normal, whereas low-likelihood sequences are detected as anomalous.

A popular Markovian approach for modeling discrete sequences is the Hidden Markov Model (HMM). Assume that a given set of discrete sequences corresponds to a normal landing flight phase. The hidden states could represent different stages that the pilot goes through in order to land the aircraft, e.g., initial descent, touch down, and braking on the runway. In any given stage, say initial descent, the pilot performs certain actions. For example, during the initial descent, the pilot reduces throttle, lowers the flaps, and uses the ailerons and elevator to stabilize the aircraft. On the other hand, in the braking stage, the pilot uses breaks as well as rudder to keep the aircraft in the middle of the runway. In HMM, the hidden states can correspond to these stages, and the actions taken in these stages are observable.

One drawback of the standard HMM as a model for pilot actions is that it encourages fast hidden state switches [57], i.e., the probability of staying in the same state decreases geometrically fast. Subsequent pilot actions usually have a time interval, and such intervals constitute normal behavior. HMM is incapable of directly handling such intervals, and it is difficult to capture such intervals as being in the same latent state for prolonged periods of time due to the geometric distribution of persistence in the same state.

To address the above limitation of HMM, we propose to use Hidden Semi-Markov Model (HSMM), which is shown in Figure 2.1, where $x_t \in \{1, \ldots, n_x\}$ represents a hidden state and $o_t \in \{1, \ldots, n_o\}$ represents an observation at time step t. Moreover, HSMM introduces one additional hidden variable, $d \in \{1, ..., n_d\}$, which controls the duration of the hidden state x.

The operation of HSMM can be described as follows. Assume that $d_{t-1} = 1$, then at time step t the hidden variable x transitions to a new state, according to the state transition distribution $p(x_t|x_{t-1})$. The time duration of this state is determined by $p(d_t|x_t)$, and the observation is drawn according to $p(o_t|x_t)$. For the subsequent time steps, the state x_t remains the same as long as $d_t \geq 1$. Once the duration counter is decreased so that $d_t = 1$, a transition to a new state is made. As is clear from the specification, HSMM are inherently capable of modeling persistence in a latent state,

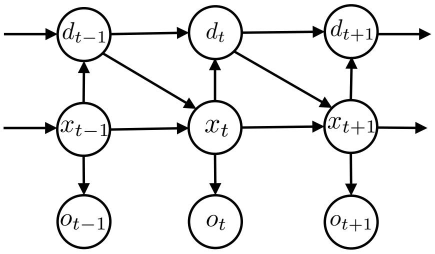


Figure 2.1: Modeling discrete data using hidden semi-Markov model (HSMM).

which is suitable for human action modeling.

The process of detecting anomalies can be described as follows (see Figure 2.2 for an illustration). Consider a database of discrete sequences, $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$, each representing a series of actions taken by a pilot during *normal* operations of an airplane. In the database, the i-th sequence of length T_i, $i = 1, \ldots, N$ is denoted as $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}$. To determine if a test sequence $\mathbf{S}^{test} = \{o_1^{test}, \ldots, o_T^{test}\}$ is anomalous, we compute the anomaly score based on the joint likelihood of the observed data. In practice, to avoid numeric underflow problems and to make this measure independent of the sequence length T, we use the normalized log-likelihood:

$$\mathcal{L}(\mathbf{S}^{test}) = \frac{\log p(\mathbf{S}^{test})}{T} = \frac{\log p(o_1^{test}, \ldots, o_T^{test})}{T}. \tag{2.1}$$

To determine specific locations in the sequence which make it anomalous and to detect anomalies in an action stream, we use the conditional probability of the current action o_t, given the observations received so far, i.e.,

$$\mathcal{M}(o_t^{test}) = p(o_t^{test} | o_1^{test}, \ldots, o_{t-1}^{test}). \tag{2.2}$$

From a graphical model perspective, HSMM is specified by three conditional probability tables (CPTs): the observation/emission probability $p(o_t|x_t)$ and the state transition and the duration probabilities given by:

$$p(d_t|x_t, d_{t-1}) = \begin{cases} p(d_t|x_t) & \text{if } d_{t-1} = 1 \\ \delta(d_t, d_{t-1}-1) & \text{if } d_{t-1} > 1 \end{cases} \tag{2.3}$$

$$p(x_t|x_{t-1}, d_{t-1}) = \begin{cases} p(x_t|x_{t-1}) & \text{if } d_{t-1} = 1 \\ \delta(x_t, x_{t-1}) & \text{if } d_{t-1} > 1 \end{cases}, \tag{2.4}$$

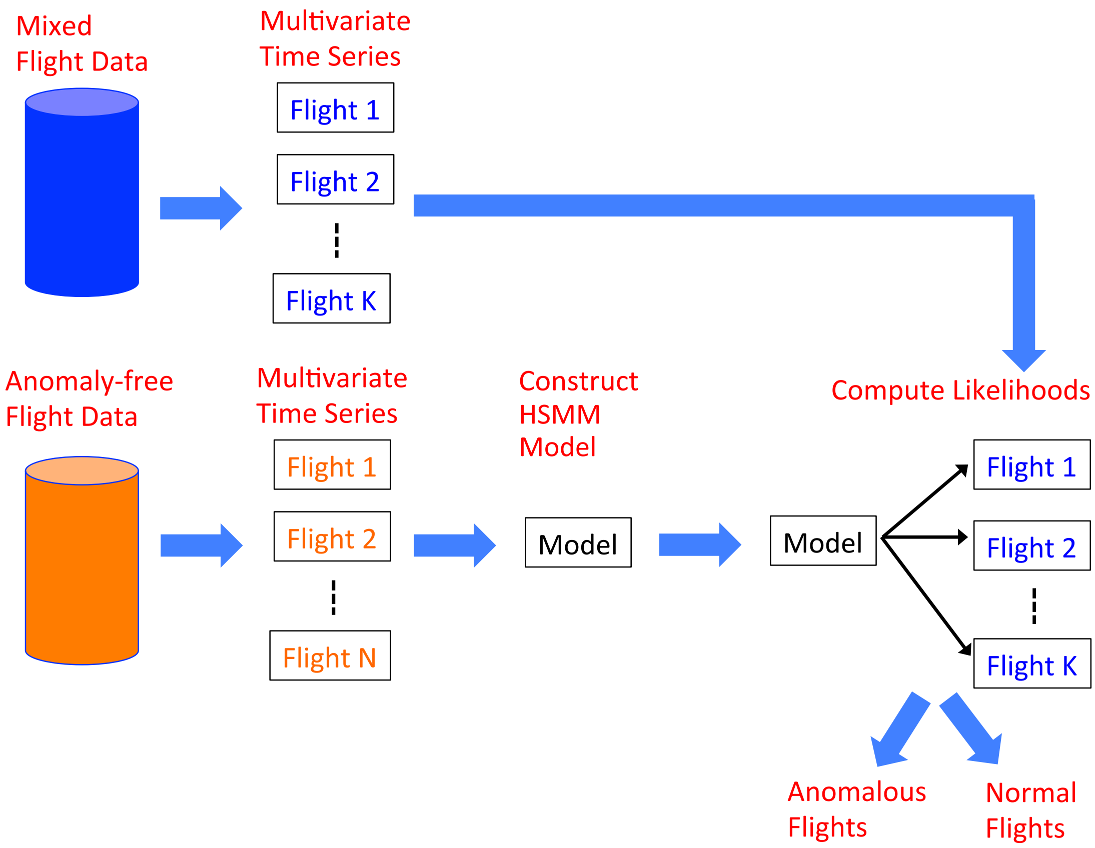


Figure 2.2: Anomaly detection framework using HSMM modeling.

where $\delta(a, b)$ denotes the Dirac delta function: $\delta(a, b) = 1$ if $a = b$ and 0 otherwise. In addition, one can consider suitable prior probabilities $p(x_0)$ and $p(d_0)$.

As we showed in (2.1), the anomaly detection procedure requires a computation of the log-likelihood of data, $\log p(o_1^{test}, \dots, o_T^{test})$ (or, equivalently, the conditional probability $p(o_t^{test}|o_1^{test}, \dots, o_{t-1}^{test})$). Traditionally, these quantities were computed using a two-stage approach. First, the HSMM parameters are estimated using methods which usually follow the initial idea due to Rabiner [57] based on the modifications of the Baum-Welch algorithm [61], which are all variants of the expectation maximization (EM) framework, presented in [18]. Once the parameters are estimated, we can then perform inference using, e.g., the forward-backward algorithm of Yu et al. [62]. However, since EM, in general, has no guarantees in estimating the parameters correctly and can suffer from slow convergence, such methods can be inefficient and/or inconsistent. The focus of our work is to develop a provably correct spectral algorithm for computing $p(\mathbf{S}^{test})$.

2.3 Spectral Algorithm for Inference

The key problems which need to be addressed in using HSMM is *learning*, i.e., estimating model parameters and *inference*, i.e., computing the probability of an observed and latent variable sequence. The methods proposed for learning are usually a variant of the expectation maximization (EM) [18] algorithm. Then, for example, a forward-backward algorithm of [62] can be used to perform inference. However, since EM has no guarantees in estimating parameters correctly, has slow convergence and whose accuracy depends on the initialization, such methods can be inefficient and inconsistent.

In recent years, there has been an increased interest in spectral algorithms, which provide computationally efficient, local-minimum-free, provably consistent algorithms for parameter estimation and/or inference. For example, Anandkumar et al. [63], [64] have proposed spectral methods for learning the parameters of a wide class of tree-structured latent graphical models, including Gaussian mixture models, topic models, and latent Dirichlet allocation. Hsu et al. [65] have proposed an efficient spectral algorithm for inference in HMMs. The algorithm learns a so called observable representation and uses it to do inference on observable variables. The approach, however, was specific to HMMs and not easily extendable to other latent variable graphical models. Parikh et al. [66] have introduced a spectral algorithm to perform inference in latent tree graphical models with arbitrary topology, and later in [67] a general spectral inference framework for latent junction trees.

We utilize the framework of [67] and introduce a novel spectral algorithm for inference in HSMMs. Since we address a more specific problem than [67], our results shed more light into the details of the spectral framework for HSMMs, allow for a sharper analysis, and yield a significantly more efficient algorithm than the general framework in [67].

There are two main technical contributions in our development of spectral algorithm for HSMM. First, by exploiting the *homogeneity* of HSMMs we make our algorithm more efficient and accurate than if we directly follow the recipe in [67] for general graphs. In particular, our approach ensures that the number of matrix multiplications and inverses is fixed and independent of sequence length. Second, we show that the order of tensors in estimated observable representation depends only *logarithmically* on the maximum length of latent state persistence.

In what follows, we introduce notations in Section 2.3.1. In Sections 2.3.2 and 2.3.4 we present HSMM inference from a tensor product perspective. Finally, in Sections 2.3.6 and 2.3.7 two versions (basic and efficient) of the spectral algorithm are presented and their properties are discussed.

2.3.1 Notations

In this section, we cover basic facts about tensor algebra, a detailed tutorial on tensors can be found in [68] or [69]. A tensor is defined as a multidimensional array of data, which will be denoted by boldface Euler script letters, e.g., $\underset{m_1,\ldots,m_N}{\boldsymbol{\mathcal{X}}} \in \mathbb{R}^{I_{m_1}\times\cdots\times I_{m_N}}$, which is N-mode tensor of dimensions $I_{m_1}\times\cdots\times I_{m_N}$. A specific mode is denoted by the subscript variable m_i, whose dimension is I_{m_i}.

Any tensor can be matrisized (or flattened) into a matrix. This mapping can be done in multiple ways, the only requirement is that the number of elements is preserved and the mapping is one-to-one. If we split the modes into two disjoint sets, one corresponding to rows and the other to columns, e.g., $\{m_1,\ldots,m_N\} = \{p_1,\ldots,p_K\}\cup\{q_1,\ldots,q_L\}$, then a matrisization of $\boldsymbol{\mathcal{X}}$ is denoted by a corresponding capital boldface letter, e.g., $\underset{p_1,\ldots,p_K q_1,\ldots,q_L}{\mathbf{X}} \in \mathbb{R}^{I_{p_1}\cdots I_{p_K}\times I_{q_1}\cdots I_{q_L}}$.

Tensor Multiplication Multiplication of two tensors is performed along specific modes. For this, we flatten each tensor to a matrix, perform the usual matrix multiplication and transform the result back to a tensor. The multiplication is denoted by a symbol × with an optional subscript representing the modes along which the operation is performed, e.g.,:

$$\underset{p_1,\ldots,p_K,r_1,\ldots,r_M}{\boldsymbol{\mathcal{Z}}} = \underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\boldsymbol{\mathcal{X}}} \times_{q_1,\ldots,q_L} \underset{q_1,\ldots,q_L,r_1,\ldots,r_M}{\boldsymbol{\mathcal{Y}}},$$

where $\underset{q_1,\ldots,q_L,r_1,\ldots,r_M}{\boldsymbol{\mathcal{Y}}} \in \mathbb{R}^{I_{q_1}\times\cdots\times I_{q_L}\times I_{r_1}\times\cdots\times I_{r_M}}$ and the resulting tensor on the left hand side is of the form $\underset{p_1,\ldots,p_K,r_1,\ldots,r_M}{\boldsymbol{\mathcal{Z}}} \in \mathbb{R}^{I_{p_1}\times\cdots\times I_{p_K}\times I_{r_1}\times\cdots\times I_{r_M}}$. Observe that in the above, we can flatten the tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ in multiple different ways as long as the matrix multiplication remains valid. For example, we could assign the multiplication modes in both tensors to columns, in this case the matrix product becomes $\mathbf{Z} = \mathbf{X}\mathbf{Y}^T$. Alternatively, the tensor $\boldsymbol{\mathcal{Y}}$ could be matrisized with the multiplication modes corresponding to rows, resulting in the product $\mathbf{Z} = \mathbf{X}\mathbf{Y}$.

An important fact about tensor multiplication is that in a series of tensor multiplications the order is irrelevant as long as the multiplication is performed along the matching modes, e.g,

$$\underset{sp}{\boldsymbol{\mathcal{X}}} \times_s \left(\underset{tr}{\boldsymbol{\mathcal{Y}}} \times_r \underset{rs}{\boldsymbol{\mathcal{Z}}}\right) = \left(\underset{sp}{\boldsymbol{\mathcal{X}}} \times_s \underset{rs}{\boldsymbol{\mathcal{Z}}}\right) \times_r \underset{tr}{\boldsymbol{\mathcal{Y}}}.$$

If we let the matrisized tensors to be $\mathbf{X} \in \mathbb{R}^{I_p \times I_s}$, $\mathbf{Y} \in \mathbb{R}^{I_t \times I_r}$ and $\mathbf{Z} \in \mathbb{R}^{I_r \times I_s}$, then the above can be verified to be true since

$$\mathbf{X}\left(\mathbf{Y}\mathbf{Z}\right) = \left(\mathbf{X}\mathbf{Z}^T\right)\mathbf{Y}^T.$$

Note that to reduce clutter, in many places we will drop the multiplication subscripts. The implied modes of multiplication can then be inferred from the subscripts of the tensors. Specifically, when two tensors are multiplied, we first check their modes and then multiply along the modes which are common to both of them. For example, in the product $\underset{pqr}{\boldsymbol{\mathcal{X}}} \times \underset{qsr}{\boldsymbol{\mathcal{Y}}}$, the implied multiplication is performed along the common modes, i.e., q and r.

Tensor Inversion We also discuss the operation of tensor inversion. Tensor inverse $\boldsymbol{\mathcal{X}}^{-1}$ is always defined with respect to a certain subset of modes and can be written as follows:

$$\underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\boldsymbol{\mathcal{X}}} \times_{q_1,\ldots,q_L} \underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\boldsymbol{\mathcal{X}}^{-1}} = \underset{p_1,\ldots,p_K,p_1,\ldots,p_K}{\boldsymbol{\mathcal{I}}},$$

where the inversion is performed along the modes $q_1, \ldots, q_L$, and $\underset{p_1,\ldots,p_K,p_1,\ldots,p_K}{\boldsymbol{\mathcal{I}}}$ denotes an identity tensor, whose elements are everywhere zero, except $\boldsymbol{\mathcal{I}}(i_1, \ldots, i_K, i_1, \ldots, i_K) = 1$. To perform inversion, we first convert tensor to a matrix, i.e., matrisize tensor. If the modes to be inverted along are associated with columns of the matrix, we compute the right matrix inverse, so that these modes get eliminated after the product. Otherwise, if those modes associated with rows, we compute left matrix inverse. Obviously, for the full rank square matrices both choices would produce the same result. For example, in the above equation the matrisized tensor might be of the form

$\underset{p_1,\dots,p_K q_1,\dots,q_L}{\mathbf{X}} \in \mathbb{R}^{I_{p_1}\cdots I_{p_K}\times I_{q_1}\cdots I_{q_L}}$, therefore, we would compute the right matrix inverse so that the modes $q_1, \dots, q_L$ are eliminated. If the matrisized $\mathbf{X}$ has full row rank, then the inverse can be computed, otherwise we could only compute its pseudo-inverse. Tensorizing the matrix $\mathbf{X}^{-1}$ gives us the desired tensor inverse.

Mode Duplication Observe that in the above, the tensor $\underset{p_1,\dots,p_K,p_1,\dots,p_K}{\mathfrak{I}}$ has duplicate modes. In general, if a tensor has duplicate modes, the corresponding sub-tensor can be interpreted as a hyper-diagonal. For example, if for a tensor $\underset{pq}{\mathfrak{X}}$ we construct a tensor $\underset{pppq}{\overline{\mathfrak{X}}}$, which has its mode p duplicated three times, then for a fixed index i, the sub-tensor $\overline{\mathfrak{X}}(:,:,:,i)$ is a hypercube with elements $\mathfrak{X}(:,i)$ on the diagonal.

Mode duplication enables us to multiply several tensors along the same mode. For example, if we need to multiply tensors $\underset{sp}{\mathfrak{X}}$, $\underset{pr}{\mathfrak{Y}}$ and $\underset{tp}{\mathfrak{Z}}$ along the mode p, then a simple product of the form

$$\underset{sp}{\mathfrak{X}} \times_p \underset{pr}{\mathfrak{Y}} \times_p \underset{tp}{\mathfrak{Z}}$$

cannot be done since any product of two tensors along the mode p would eliminate it, preventing any further multiplications. In general, if there are N multiplications along the specific mode, then there are must be cumulatively $2N$ number of times such a mode is encountered in the participating tensors. In our example, we might duplicate the mode p in, say, tensor $\mathfrak{Z}$ to have

$$\underset{sp}{\mathfrak{X}} \times_p \left(\underset{pr}{\mathfrak{Y}} \times_p \underset{tpp}{\mathfrak{Z}} \right),$$

so that there are two multiplications over mode p and cumulatively there are four times such a mode is encountered in the participating tensors. To reduce clutter, we sometimes do not explicitly show the duplicated variables in the subscripts; the implied mode repetition will be evident from the context or explicitly stated in cases when there is a confusion. For example, the identity tensor will often be written as $\underset{p_1,\dots,p_K}{\mathfrak{I}}$.

2.3.2 Problem Formulation

We start by considering the matrix forms of the HSMM parameters and writing the computations in tensor notation, as introduced in Section 2.3.1. Specifically, $p(d_t|x_t, d_{t-1} = 1)$ is denoted as $D \in \mathbb{R}^{n_d \times n_x}$, $p(x_t|x_{t-1}, d_{t-1} = 1)$ is denoted as $X \in \mathbb{R}^{n_x \times n_x}$, and

$p(o_t|x_t)$ as $O \in \mathbb{R}^{n_o \times n_x}$. We make the following assumptions on the HSMM parameters:

Assumptions

$A1$. $\mathcal{X}$ is full rank and has non-zero probability of visiting any state from any other state.

$A2$. D has a non-zero probability of any duration in any state.

$A3$. O is full column rank and, as a consequence, $n_x \leq n_o$.

We provide some comments on the above assumptions. We note that the assumption $A1$ can be relaxed to allow zero entries (while still ensuring full rank structure) and thus prevent certain states to be directly reachable from other states; however, this would require more involved analysis based on the mixing time of the corresponding Markov chain [70], and is not pursued in this work. Also, observe that the assumption of $n_x \leq n_o$ is needed in order to ensure that hidden states are identifiable, although recent work is showing that such an assumption can be relaxed in some cases [71]. Intuitively, it means that the number of different observations coming from each state is large enough, so that one hidden state can be differentiated from the other.

To express the joint probability $p(o_1, \ldots, o_T)$ for any possible observation sequence in tensor form, we utilize the junction tree algorithm [7]. The resulting tree is shown in Figure 2.3 and it corresponds to the graphical model of HSMM in Figure 2.1. Recall, that the junction tree is a tree-structured representation of an arbitrary graph enabling efficient inference. It can be constructed by forming a maximal spanning tree from the cliques of the graph. The cliques then represent vertices in the junction tree and the edges connecting the vertices are labeled with variables common to two cliques it connects. The set of variables on the edges are referred to as separators. For example, in Figure 2.3 the cliques $\mathbb{X}_t$ and $\mathbb{D}_t$ have two variables in common, x_{t-1} and d_{t-1}, and which define the sepatator between $\mathbb{X}_t$ and $\mathbb{D}_t$.

We proceed by representing the clique CPTs of the junction tree as tensors. For example, the clique $\mathbb{X}_t$, containing the CPT of $p(x_t|x_{t-1}, d_{t-1})$ is represented as tensor $\underset{x_t|x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{X}}}$. For ease of exposition, the tensor's modes are named based on the variables on which the tensor depends. We also keep the conditioning symbol $|$, for clarity. Similarly, we represent the clique $\mathbb{D}_t$ with its CPT $p(d_t|x_t, d_{t-1})$ as tensor $\underset{d_t|x_t d_{t-1}}{\boldsymbol{\mathcal{D}}}$, and

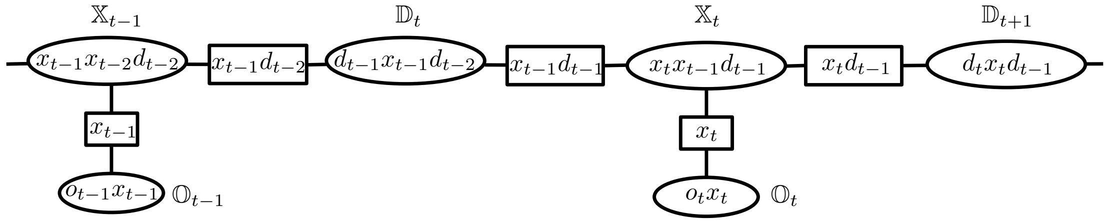


Figure 2.3: Junction Tree for Hidden Semi-Markov Model. The ovals represent cliques, which are denoted by capital blackboard bold variables; the rectangles denote separators. Symbols within the shapes represent the variables on which the corresponding potentials depend.

$\mathbb{O}_t$ containing $p(o_t|x_t)$ as tensor $\underset{o_t|x_t}{\boldsymbol{\mathcal{O}}}$.

If we denote the joint probability of the observed sequence $p(o_1, \ldots, o_T)$ as $\underset{o_1,\ldots,o_T}{\boldsymbol{\mathcal{P}}}$ then the message passing for the junction tree algorithm in Figure 2.3 can be represented as tensor multiplications:

$$\underset{o_1,\ldots,o_T}{\boldsymbol{\mathcal{P}}} = \prod_t \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{D}}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\boldsymbol{\mathcal{X}}} \times_{x_t} \underset{o_t|x_t}{\boldsymbol{\mathcal{O}}} \right), \tag{2.5}$$

where, for simplicity, we denoted by $\prod_t$ the tensor product over multiple time steps.

Note that in (2.5) the neighboring tensors are multiplied along the modes which are the separator variables between two corresponding neighboring cliques in Figure 2.3. Therefore, as we discussed in Section 2.3.1, if a certain mode of a tensor is to participate multiple times in products with other tensor, the mode must be duplicated for the expression to remain correct. It can easily be seen from the junction tree that the number of times the mode is duplicated depends on the number of times such a variable appears in separators adjacent to the clique. For example, the tensor $\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\boldsymbol{\mathcal{X}}}$ has a mode x_{t-1} appearing once in the separator connecting $\mathbb{X}_t$ and $\mathbb{D}_t$ in Figure 2.3, while x_t appears a total of two times - once in the separator connecting $\mathbb{X}_t$ and $\mathbb{O}_t$, and once in the separator connecting $\mathbb{X}_t$ and $\mathbb{D}_{t+1}$. Finally, d_{t-1} appears in the separator between $\mathbb{D}_t$ and $\mathbb{X}_t$, and between $\mathbb{D}_{t+1}$ and $\mathbb{X}_t$. Applying the same reasoning to tensors $\boldsymbol{\mathcal{D}}$ and $\boldsymbol{\mathcal{O}}$ results in the expression (2.5).

2.3.3 Summary of Results

In this work, we represent expression (2.5), which is defined in terms of unknown model parameters, in a different form, called observable representation, where all the factors can be estimated directly from data using certain sample moments without knowledge of model parameters. Such an observable form is derived in Sections 2.3.4 and 2.3.5. Based on the obtained representation, we propose in Section 2.3.6 a simple spectral algorithm, which requires estimating $\mathfrak{X}$, $\mathfrak{D}$ and $\mathfrak{O}$ for all the time stamps t. This estimation process is expensive as it involves costly tensor operations to be performed at each time index t. Moreover, the accurate estimation of these tensors requires large number of training sequences which might not be available, leading to inaccurate and unstable computations. However, exploiting the homogeneity property of HSMMs, i.e., the fact that the probability distributions, which the above tensors represent, are independent of time index t, we derive computationally more efficient and accurate spectral algorithm in Section 2.3.7 requiring estimation of only three tensors for all the time stamps t. Although the computational complexity of inference, i.e., the evaluation of expression (2.5), is not affected by the introduced modifications, the overall algorithm becomes faster and more accurate. In Section 2.4 we return to the results of Sections 2.3.4 and establish the conditions under which the derived observable representation exists. In particular, our analysis shows that the number of dimensions of the required sample moments has logarithmic dependence on the longest state persistence n_d. Such conclusion is in contrast to the analysis, which would follow from the work of [67], in which case the required number of dimensions in the estimated sample moments would have had linear dependence on n_d. The exponential reduction in the size of the sample moments represents significant improvement in algorithm's efficiency and accuracy. Finally, we evaluated the proposed algorithm using synthetic and real datasets and compared its performance with the traditional EM approach. The main conclusion from such evaluations is that for large enough datasets the spectral method gets similar or better performance than EM, while at the same time being orders of magnitude faster than EM.

2.3.4 Observable Form Representation

Observe that the computation of the joint probability in (2.5) requires knowledge of the unknown model parameters. Our goal is to change the tensor representation such that $\underset{o_1,\ldots,o_T}{\mathfrak{P}}$ can be written in terms of the quantities directly computable from data. To that end, we follow [67] and between every two factors in (2.5) introduce an identity tensor with the modes corresponding to the modes along which the multiplication is performed. For example, consider a part of (2.5) after introducing identity tensors:

$$\begin{aligned} &\times \underset{x_{t-1}d_{t-2}}{\mathfrak{I}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathfrak{D}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathfrak{I}} \times_{x_{t-1}d_{t-1}} \\ &\qquad \times_{x_{t-1}d_{t-1}} \left(\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\mathfrak{X}} \times_{x_t} \underset{x_t}{\mathfrak{I}} \times_{x_t} \underset{o_t x_t}{\mathfrak{O}} \right) \times_{x_t d_{t-1}} \underset{x_t d_{t-1}}{\mathfrak{I}} \times, \end{aligned} \tag{2.6}$$

where all the identity tensors have duplicated modes which are not shown.

Now rewrite each of the identity tensors in (2.6) as a multiplication of some factor times its inverse. For example,

$$\underset{x_t}{\mathfrak{I}} = \underset{\omega_{x_t}x_t}{\mathfrak{F}} \times_{\omega_{x_t}} \underset{\omega_{x_t}x_t}{\mathfrak{F}^{-1}},$$

for some invertible factor $\underset{\omega_{x_t}x_t}{\mathfrak{F}}$, whose modes are x_t and ω_{x_t}. Note that the choice of mode x_t is fixed and is determined by the modes of the identity tensor $\underset{x_t}{\mathfrak{I}}$, while the mode ω_{x_t} is not fixed and we have a freedom in selecting it. Moreover, observe that since the tensor inversion is done along the mode ω_{x_t} and the matrix $\mathbf{F}$ has its rows associated with mode ω_{x_t}, we need to ensure such a matrix has full column rank for the inverse to exist and for the product $\mathbf{F}^{-1}\mathbf{F}$ to be the identity matrix (see Section **??** for more details on tensor inversion). Based on the above discussion, we choose tensor $\mathfrak{F}$ such that (i) ω_{x_t} are the observed variables, (ii) $\underset{\omega_{x_t}x_t}{\mathfrak{F}}$ is invertible and (iii) we interpret the factor $\underset{\omega_{x_t}x_t}{\mathfrak{F}}$ as corresponding to a conditional probability distribution, i.e., $p(\omega_{x_t}|x_t)$ and therefore write $\underset{\omega_{x_t}|x_t}{\mathfrak{F}}$.

After expanding each of the identity tensors, regrouping the factors and recalling that in a series of tensor multiplication the order is irrelevant, we can identify three modified tensors:

$$\begin{aligned}
\underset{\omega_{x_{t-1}d_{t-2}}\omega_{x_{t-1}d_{t-1}}}{\tilde{\mathcal{D}}} &= \underset{\omega_{x_{t-1}d_{t-2}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\mathcal{F}} \\
\underset{\omega_{x_{t-1}d_{t-1}}\omega_{x_t}\omega_{x_td_{t-1}}}{\tilde{\mathcal{X}}} &= \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{\omega_{x_t}|x_t}{\mathcal{F}} \right) \times_{x_td_{t-1}} \underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\mathcal{F}} \\
\underset{\omega_{x_t}o_t}{\tilde{\mathcal{O}}} &= \underset{\omega_{x_t}|x_t}{\mathcal{F}^{-1}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}} .
\end{aligned}$$

Note that although each of the above tensors depends only on the observed variables ω, how to estimate them is not clear yet: the expressions on the right depend on the unknown model parameters, while the tensors on the left do not correspond to valid probability distributions (due to the presence of inverses $\mathcal{F}^{-1}$), and so cannot be estimated from data using sample moments. For example, $\underset{\omega_{x_{t-1}d_{t-2}}\omega_{x_{t-1}d_{t-1}}}{\tilde{\mathcal{D}}}$ is not a tensor form of $p(\omega_{x_{t-1}d_{t-2}}, \omega_{x_{t-1}d_{t-1}})$.

Next, we discuss the choice of the observable set ω in the factors $\mathcal{F}$. From Figure 2.3 we can see that there are three types of separators which depend on $x_{t-1}d_{t-1}$, x_td_{t-1} and x_t, consequently, there are three types of identity tensors which we introduced in (2.6), i.e., $\underset{x_{t-1}d_{t-1}}{\mathcal{I}}$, $\underset{x_td_{t-1}}{\mathcal{I}}$ and $\underset{x_t}{\mathcal{I}}$. Therefore, we need to define three types of observable sets $\omega_{x_{t-1}d_{t-1}}$, $\omega_{x_td_{t-1}}$ and ω_{x_t}. There could be multiple choices for these sets, one of them is $\omega_{x_{t-1}d_{t-1}} = \omega_{x_td_{t-1}} = \{o_{t+1}, o_{t+2}, \ldots\}$ for all t (see Figure 2.4 for an illustration). Ideally, we want these sets to be of minimal size, since they need to be estimated from observations. The detailed description of how many and which of these observations to select to get a minimal set is deferred until Section 2.4, where we also show that we can set $\omega_{x_t} = o_t$.

In what follows, we define $\mathsf{\mathbf{O}}_{R_t} := \{o_{t+1}, o_{t+2}, \ldots\}$, to emphasize that this is a fixed set of observations whose length is yet to be determined, starting after time stamp t and going to the right (or forward in time) in the graphical model in Figure 2.1. With these definitions, setting $\omega_{x_{t-1}d_{t-1}} = \mathsf{\mathbf{O}}_{R_t}$, $\omega_{x_td_{t-1}} = \mathsf{\mathbf{O}}_{R_t}$, $\omega_{x_{t-1}d_{t-2}} = \mathsf{\mathbf{O}}_{R_{t-1}}$ and $\omega_{x_t} = o_t$, we can now rewrite (2.5) in the form:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \prod_t \underset{\mathsf{\mathbf{O}}_{R_{t-1}}\mathsf{\mathbf{O}}_{R_t}}{\tilde{\mathcal{D}}} \times_{\mathsf{\mathbf{O}}_{R_t}} \left(\underset{\mathsf{\mathbf{O}}_{R_t}o_t\mathsf{\mathbf{O}}_{R_t}}{\tilde{\mathcal{X}}} \times_{o_t} \underset{o_to_t}{\tilde{\mathcal{O}}} \right) . \tag{2.7}$$

Comparing (2.5) and (2.7) we see that the above equation expresses the joint probability distribution in the observable form. As noted above, we cannot yet use this formula

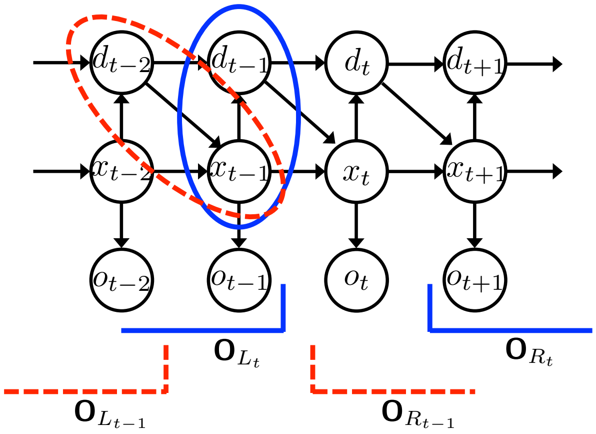


Figure 2.4: Conditional independence in HSMM. The figure depicts two sets of relationships: $\mathbf{O}_{L_t}$ and $\mathbf{O}_{R_t}$ are independent conditioned on $x_{t-1}d_{t-1}$, similarly, $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ are conditionally independent given $x_{t-1}d_{t-2}$. We defined $\mathbf{O}_{L_t} = \{\dots, o_{t-2}, o_{t-1}\}$ and $\mathbf{O}_{R_t} = \{o_{t+1}, o_{t+2}, \dots\}$.

in practice since we do not know how to compute the transformed tensors. In what follows, we show how to estimate such tensors directly from data, without the need for the model parameters.

2.3.5 Estimation of Observable Tensors

In this Section we express each of the tensors in (2.7) in the form suitable for estimation directly from the observed sequences.

Computation of Tensor $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathfrak{D}}}$

Consider the tensor from Section 2.3.4

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathfrak{D}}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathfrak{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathfrak{F}}, \tag{2.8}$$

whose modes are the observable variables $\mathbf{O}_{R_{t-1}}$ and $\mathbf{O}_{R_t}$. To estimate this tensor from data, consider $\mathbf{O}_{L_{t-1}}$, a set of the observed variables such that $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ are independent, conditioned on $x_{t-1}d_{t-2}$ (see Figure 2.4):

$$p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_{t-1}}) = \sum_{x_{t-1}d_{t-2}} p(\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2})p(\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2})p(x_{t-1}d_{t-2}). \tag{2.9}$$

The above conditional independence relationship can be written in tensor form:

$$\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{K}}}, \tag{2.10}$$

where tensor $\boldsymbol{\mathcal{K}}$ represents the marginal $p(x_{t-1}, d_{t-2})$. Note that, though not shown, the modes x_{t-1} and d_{t-2} need to appear twice in $\boldsymbol{\mathcal{K}}$, since it interacts with both other terms (see the discussion on mode duplication in Section 2.3.1). The set $\mathbf{O}_{L_{t-1}}$ is defined in a way similar to $\mathbf{O}_{R_t}$ but with the set of observations starting at time stamp $t-2$ and going to the left (or backward in time), i.e., $\mathbf{O}_{L_{t-1}} := \{\ldots, o_{t-3}, o_{t-2}\}$ (see Figure 2.4).

Next, we express the inverse of the tensor $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}}$ from (2.10) and substitute back to (2.8). For this, we observe that in (2.8) the tensor $\boldsymbol{\mathcal{F}}^{-1}$ is inverted with respect to mode $\mathbf{O}_{R_{t-1}}$, therefore, we do the following:

$$\begin{aligned}
\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}} \times_{\mathbf{O}_{R_{t-1}}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}^{-1}} &= \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{I}}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{K}}} \\
\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}^{-1}} &= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{K}}}, \qquad (2.11)
\end{aligned}$$

where $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}^{-1}}$ is inverted with respect to mode $\mathbf{O}_{L_{t-1}}$. Next, substituting (2.11) back to (2.8), we get

$$\begin{aligned}
\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\boldsymbol{\mathcal{D}}}} &= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \overbracket{\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{K}}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{D}}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{F}}}} \\
&= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\boldsymbol{\mathcal{M}}}, \qquad (2.12)
\end{aligned}$$

where we have eliminated all the latent variables by multiplying the last four terms on the first line.

Observe that the tensors $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}}$ and $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\boldsymbol{\mathcal{M}}}$ represent valid joint probability distributions over a subset of observations $p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_{t-1}})$ and $p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_t})$, respectively, and though they are defined with respect to unknown model parameters (as, for example, in (2.9)), we can readily estimate them from data. For example, $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\boldsymbol{\mathcal{M}}}$ is a tensor, where each entry is computed from the frequency of co-occurrence of tuples of the observed symbols $\{\ldots, o_{t-3}, o_{t-2}, o_{t+1}, o_{t+2}, \ldots\}$. Ideally, we want a small number

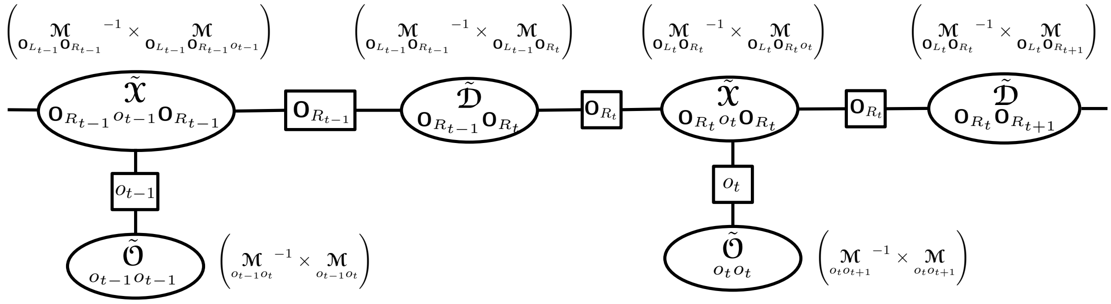

$\left(\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathcal{M}}^{-1} \times \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}o_{t-1}}{\mathcal{M}}\right)$
$\underset{\mathbf{o}_{R_{t-1}}o_{t-1}\mathbf{o}_{R_{t-1}}}{\tilde{\mathcal{X}}}$
$\mathbf{o}_{R_{t-1}}$
$\left(\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathcal{M}}^{-1} \times \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathcal{M}}\right)$
$\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathcal{D}}}$
$\mathbf{o}_{R_t}$
$\left(\underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}}^{-1} \times \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}o_t}{\mathcal{M}}\right)$
$\underset{\mathbf{o}_{R_t}o_t\mathbf{o}_{R_t}}{\tilde{\mathcal{X}}}$
$\mathbf{o}_{R_t}$
$\left(\underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}}^{-1} \times \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_{t+1}}}{\mathcal{M}}\right)$
$\underset{\mathbf{o}_{R_t}\mathbf{o}_{R_{t+1}}}{\tilde{\mathcal{D}}}$
o_{t-1}
o_t
$\underset{o_{t-1}o_{t-1}}{\tilde{\mathcal{O}}}$
$\left(\underset{o_{t-1}o_t}{\mathcal{M}}^{-1} \times \underset{o_{t-1}o_t}{\mathcal{M}}\right)$
$\underset{o_t o_t}{\tilde{\mathcal{O}}}$
$\left(\underset{o_t o_{t+1}}{\mathcal{M}}^{-1} \times \underset{o_t o_{t+1}}{\mathcal{M}}\right)$

Figure 2.5: Graphical representation of the HSMM spectral algorithm for inference in Algorithm 1. As compared to junction tree in Figure 2.3, the cliques and separators are now defined in terms of the tensors, which are defined with respect to the observed data. The expressions in the parenthesis show the observable representation of the corresponding tensors.

of observation symbols since we need to estimate their co-occurrence frequency from the training data. A precise characterization of how many and which of these symbols suffices for the analysis will be done in Section 2.4.

Computation of Tensor $\underset{\mathbf{o}_{R_t} o_t \mathbf{o}_{R_t}}{\tilde{\mathcal{X}}}$

The form of this tensor was established at the beginning of Section 2.3.5 to be:

$$\underset{\mathbf{o}_{R_t} o_t \mathbf{o}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}} \right) \times_{x_t d_{t-1}} \underset{\mathbf{o}_{R_t}|x_t d_{t-1}}{\mathcal{F}}. \tag{2.13}$$

Consider the following conditional independence relationship (see Figure 2.4):

$$\underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}} = \underset{\mathbf{o}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}}, \tag{2.14}$$

where $\underset{x_{t-1}d_{t-1}}{\mathcal{K}} = \underset{x_{t-1}d_{t-1}x_{t-1}d_{t-1}}{\mathcal{K}}$ and we omitted the duplicated modes.

We express the inverse of tensor $\underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ from the above equation

$$\underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} = \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{o}_{L_t}} \underset{\mathbf{o}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}},$$

where tensor $\underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ is inverted with respect to mode $\mathbf{o}_{R_t}$, while $\underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}}$ is inverted with respect to mode $\mathbf{o}_{L_t}$. Substituting back to (2.13), we get

$$\begin{aligned}\underset{\mathbf{o}_{R_t}o_t\mathbf{o}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{o}_{L_t}} \underset{\mathbf{o}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}} \times_{x_{t-1}d_{t-1}} \\ \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}} \right) \times_{x_td_{t-1}} \underset{\mathbf{o}_{R_t}|x_td_{t-1}}{\mathcal{F}}.\end{aligned}$$

Considering the last five factors and multiplying them together, we obtain

$$\underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}o_t}{\mathcal{M}} = \underset{\mathbf{o}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}} \right) \times_{x_td_{t-1}} \underset{\mathbf{o}_{R_t}|x_td_{t-1}}{\mathcal{F}}.$$

Finally, (2.13) can now be written as

$$\underset{\mathbf{o}_{R_t}o_t\mathbf{o}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{o}_{L_t}} \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}o_t}{\mathcal{M}}, \tag{2.15}$$

where the right hand side can now be estimated directly from data, without the need for the model parameters.

Computation of Tensor $\underset{o_to_t}{\tilde{\mathcal{O}}}$

Finally, we consider the tensor

$$\underset{o_to_t}{\tilde{\mathcal{O}}} = \underset{o_t|x_t}{\mathcal{F}^{-1}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}}. \tag{2.16}$$

The conditional independence relationship can take the form

$$\underset{o_to_{t+1}}{\mathcal{M}} = \underset{o_t|x_t}{\mathcal{F}} \times_{x_t} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}}.$$

Expressing the inverse of $\underset{o_t|x_t}{\mathcal{F}}$

$$\underset{o_t|x_t}{\mathcal{F}^{-1}} = \underset{o_to_{t+1}}{\mathcal{M}^{-1}} \times_{o_{t+1}} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}},$$

and substituting in (2.16), we get

$$\begin{aligned}\underset{o_to_t}{\tilde{\mathcal{O}}} &= \underset{o_to_{t+1}}{\mathcal{M}^{-1}} \times_{o_{t+1}} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}} \\ &= \underset{o_to_{t+1}}{\mathcal{M}^{-1}} \times_{o_{t+1}} \underset{o_to_{t+1}}{\mathcal{M}}. \end{aligned} \tag{2.17}$$

2.3.6 Basic Version of Spectral Algorithm

The basic version of the spectral HSMM algorithm to compute $\underset{o_1,\ldots,o_T}{\mathcal{P}}$ entirely using the observed variables can be described as a two step process: in the learning step, compute $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}}$, $\underset{\mathbf{O}_{R_{t-1}}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}}$ and $\underset{o_to_t}{\tilde{\mathcal{O}}}$ for each t using (2.12), (2.15) and (2.17) from the training data. In the inference step, use (2.7) to compute $p(\mathbf{S}^{test})$. Algorithm 1 shows its basic version and Figure 2.5 shows the graphical representation of this algorithm in terms of the transformed junction tree of Figure 2.3.

As an example, consider the learning step of the algorithm and the computation of tensor in (2.12), i.e.,

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}.$$

For a fixed t, we estimate each entry of $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}}$ from the frequency of co-occurrence of tuples of the observed symbols $\{\ldots, o_{t-3}, o_{t-2}, o_{t+1}, o_{t+2}, \ldots\}$ in the given dataset (the sets $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ were defined at the beginning of Section 2.3.5). Next, following our discussion after the equation (2.11), we invert $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}}$ along the modes $\mathbf{O}_{L_{t-1}}$. For this, we matrisize the tensor so that the modes $\mathbf{O}_{L_{t-1}}$ are associated with columns and $\mathbf{O}_{R_{t-1}}$ with rows in matrix $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}}$ (see Section 2.3.1 for the discussion on tensor matrisization and inversion). Finally, we compute the right inverse of the matrix to obtain $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}^{-1}}$. Similarly, we estimate the tensor $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}$ using the corresponding co-occurrences of the observed symbols. Matrisizing the result, so that the rows correspond to the modes $\mathbf{O}_{L_{t-1}}$ and the columns to $\mathbf{O}_{R_t}$, we get the matrix $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathbf{M}}$. The multiplication $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}^{-1}} \cdot \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathbf{M}} = \underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathbf{D}}}$ produces a matrix, which is then converted to a tensor to get the final result in (2.12).

In the inference step we perform tensor multiplications for each t running along the length of the testing sequence. The only nuance here is that before multiplying the tensor $\underset{o_to_t}{\tilde{\mathcal{O}}}$ with others, the second mode o_t, whose dimension is n_o is collapsed into a scalar. This operation is denoted as $\underset{o_to_t}{\tilde{\mathcal{O}}}\Big|_{o_t=o_t^{test}}$, which means that based on the value of the tth symbol in testing sequence, we select the column corresponding to the element o_t^{test}. For example, if $\underset{o_to_t}{\tilde{\mathcal{O}}} \in \mathbb{R}^{10\times 10}$ and $o_t^{test} = 3$ then $\underset{o_to_t}{\tilde{\mathcal{O}}}\Big|_{o_t=o_t^{test}} \in \mathbb{R}^{10\times 1}$, a third column in the original matrix.

Algorithm 1 Basic Spectral Algorithm for HSMM inference

```
Input: Training sequences: $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}, i = 1, \ldots, N$.
Testing sequence: $\mathbf{S}^{test} = \{o_1^{test}, \ldots, o_T^{test}\}$.
Output: $p(\mathbf{S}^{test})$

Learning phase:
for all $t$ do
  Estimate $\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}}$, $\underset{\mathbf{o}_{R_t}o_t\mathbf{o}_{R_t}}{\tilde{\mathfrak{X}}}$ and $\underset{o_to_t}{\tilde{\mathfrak{O}}}$ from data $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$ using equations (2.12), (2.15) and (2.17).
end for

Inference phase:
$p(\mathbf{S}^{test}) = 1$
for $t = T$ down to $t = 1$ do
  $p(\mathbf{S}^{test}) = p(\mathbf{S}^{test}) \times \underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}} \times_{\mathbf{o}_{R_t}} \left( \underset{\mathbf{o}_{R_t}o_t\mathbf{o}_{R_t}}{\tilde{\mathfrak{X}}} \times_{o_t} \underset{o_to_t}{\tilde{\mathfrak{O}}} \Big|_{o_t = o_t^{test}} \right)$
end for
```

Analyzing (2.12), (2.15) and (2.17), we see that the computational complexity of the learning phase of the algorithm is determined by the tensor inverses and multiplications. For example, if in (2.12) we denote $|\mathbf{o}_R| = |\mathbf{o}_L| = \ell$ (in Section 2.4 we will show that $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$), then $\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathbf{M}} \in \mathbb{R}^{n_o^\ell \times n_o^\ell}$ and $\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}} \in \mathbb{R}^{n_o^\ell \times n_o^\ell}$. The computational complexity of the multiplications and inversions would then be $\mathcal{O}(n_o^{3\ell})$. Performing this computations for all t and assuming that the length of the sequences is T, would result in $\mathcal{O}\left(n_o^{3\ell}T\right)$. Additionally, with N training examples there will be a cost of $\mathcal{O}\left(\ell NT\right)$ to estimate the sample moments $\mathfrak{M}$, which is based on counting the co-occurrences of certain observable symbols. In the inference phase of the algorithm, we perform a series of tensor multiplications with the cost of $\mathcal{O}(n_o^{3\ell}T)$.

2.3.7 Efficient Version of Spectral Algorithm

Note that for large ℓ the accurate estimation of tensors $\mathfrak{M}$ for each t will require large number of training sequences which might not be available, leading to inaccurate and unstable computations. Observe, however, that for example the estimated sample-based tensors $\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathfrak{M}}$ in (2.12) for each t estimate the same population quantity due to homogeneity of HSMM. Thus, a *novel* aspect of our work is the improvement of the accuracy and efficiency of the basic algorithm 3 by exploiting the homogeneity property of HSMM and estimating the tensors $\tilde{\mathfrak{X}}$, $\tilde{\mathfrak{D}}$ and $\tilde{\mathfrak{O}}$ in the batch, by pooling the samples across different t and then averaging the result. Thus, we compute only three tensors for all t, as opposed to computing these tensors for each t.

We show the details for computing the tensors $\tilde{\mathfrak{D}}$ in the batch form. The derivations for other tensors $\tilde{\mathfrak{X}}$ and $\tilde{\mathfrak{O}}$ can be computed in a similar manner. Recall from (2.12) the form of $\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}}$, and consider the following alternative expression, based on the sum over all t:

$$\tilde{\mathfrak{D}} = \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathfrak{M}}\right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathfrak{M}}\right), \tag{2.18}$$

where $\mathbf{o}_L$ denotes a generic mode of the averaged tensor $\mathfrak{M}$, corresponding to $\mathbf{o}_{L_{t-1}}$ for all t. Note that in practice, instead of summation, we use averaging to avoid numerical overflow problems. It is equivalent to the considered expression in (2.18), since the term $\frac{1}{T}$ then cancels out. Since

$$\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathfrak{M}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathfrak{K}}, \tag{2.19}$$

the first term inside brackets can be rewritten as:

$$\begin{aligned}
&\sum_t \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathfrak{K}} \\
&\quad \overset{(a)}{=} \sum_t \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathfrak{F}}} \\
&\quad \overset{(b)}{=} \underset{\mathbf{o}_{R_2}|x_2d_1}{\mathfrak{F}} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathfrak{F}}}\right),
\end{aligned} \tag{2.20}$$

where in (a) we combined the two factors, i.e., $\underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathfrak{F}}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}x_{t-1}d_{t-2}}{\mathfrak{K}}$ and in (b) we used the homogeneity property of HSMM, i.e., the fact that $\underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathfrak{F}}$ does not depend on time stamp t, and extracted one of the common factors, in fact, the first factor. Note that the term $\underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathfrak{F}}}$, on the other hand, does depend on t since the factor $\underset{x_{t-1}d_{t-2}}{\mathfrak{K}}$, which represents the probability $p(x_{t-1}, d_{t-2})$, changes as the time stamp t changes.

Similarly, since

$$\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathcal{M}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}, \tag{2.21}$$

rewrite the second term in (2.18) as

$$\begin{aligned}
&\sum_t \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \\
&\quad = \sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \\
&\quad = \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} \right) \times \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times_{x_2d_2} \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}},
\end{aligned} \tag{2.22}$$

where we used the transformations similar as in (2.20), i.e., the fact that the factors $\underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}}$ and $\underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ are homogeneous, independent of t. Now if we multiply the inverse of (2.20) with (2.22), we get

$$\underset{\mathbf{o}_{R_2}|x_2d_1}{\mathcal{F}^{-1}} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} \right)^{-1} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} \right) \times \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}} \tag{2.23}$$

$$\begin{aligned}
&= \underset{\mathbf{o}_{R_2}|x_2d_1}{\mathcal{F}^{-1}} \times_{x_2d_1} \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times_{x_2d_2} \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}} \\
&= \underset{\mathbf{o}_{R_2}\mathbf{o}_{R_3}}{\tilde{\mathcal{D}}} = \underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathcal{D}}},
\end{aligned} \tag{2.24}$$

where in (2.23) we used the fact that the order in which tensors are multiplied is irrelevant and also the fact that the terms in parenthesis are invertible. This is due to the fact that the set of observations $\mathbf{o}_{L_{t-1}}$ for all t is selected so as to make each of the summand invertible (see Section 2.4 for the details about the choice of $\mathbf{o}_{L_{t-1}}$). Moreover, in (2.24) we used the definition of $\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathcal{D}}}$

$$\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times \underset{d_{t-1}|x_{t-1}d_{t-2}}{\mathcal{D}} \times \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}},$$

together with the homogeneity property of HSMM.

Therefore, we can conclude that the batch form of the tensor takes the form:

$$\tilde{\mathcal{D}} = \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathcal{M}} \right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathcal{M}} \right). \tag{2.25}$$

Algorithm 2 Efficient Spectral Algorithm for HSMM inference

```
Input: Training sequences: $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}, i = 1, \ldots, N$.
Testing sequence: $\mathbf{S}^{test} = \{o_1^{test}, \ldots, o_T^{test}\}$.
Output: $p(\mathbf{S}^{test})$

Learning phase:
Estimate $\tilde{\mathfrak{D}}, \tilde{\mathfrak{X}}$ and $\tilde{\mathfrak{O}}$ from data $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$ using equations (2.25), (2.26) and (2.27).

Inference phase:
$p(\mathbf{S}^{test}) = 1$
for $i = T$ down to $i = 1$ do
    $p(\mathbf{S}^{test}) = p(\mathbf{S}^{test}) \times \tilde{\mathfrak{D}} \times \left(\tilde{\mathfrak{X}} \times \tilde{\mathfrak{O}}|_{o=o_i^{test}}\right)$
end for
```

Similar derivations can be carried out to obtain the rest of the tensors in the batch form:

$$\tilde{\mathfrak{X}} = \left(\sum_t \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t}}{\mathfrak{M}}\right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_t}\mathbf{o}_{R_t} o_t}{\mathfrak{M}}\right) \tag{2.26}$$

$$\tilde{\mathfrak{O}} = \left(\sum_t \underset{o_t o_{t+1}}{\mathfrak{M}}\right)^{-1} \times_o \left(\sum_t \underset{o_t o_{t+1}}{\mathfrak{M}}\right). \tag{2.27}$$

where in the last expression the mode o corresponds to the mode $o_{t_{t+1}}$ after averaging of tensor $\underset{o_t o_{t+1}}{\mathfrak{M}}$ for all t.

Analyzing (2.25), (2.26) and (2.27), we see that the computational complexity of the learning phase of the algorithm is now $\mathcal{O}\left((n_o^{2\ell} + \ell N)T\right)$, mainly determined by the tensor additions and the estimation of the sample moments $\mathfrak{M}$. The number of inverses and multiplications is now fixed and independent of sequence length T. Specifically, there will be three tensor multiplications and inversions for a total cost of $\mathcal{O}(n_o^{3\ell})$. The computational complexity of the inference phase is $\mathcal{O}(n_o^{3\ell}T)$, which is the same as for Algorithm 1.

Note that such a batch tensor computation significantly improves the accuracy of the resulting spectral algorithm. In part, this is due to the fact that we now use more data to estimate the tensors as compared to the original form (2.7). The estimates obtained in this form have lower variance, which in turn ensures that the inverses we compute in (2.25), (2.26) and (2.27) are more stable and accurate.

2.4 Rank Analysis of Observable Tensors

In this Section we present a careful technical analysis to establish logarithmic dependence of the number of modes in the tensor on maximum latent state persistence. Recall that in Section 2.3.5, when we derived the equations (2.12), (2.15) and (2.17), we glossed over the question of the existence of tensor inverses $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}}$, $\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}^{-1}}$ and $\underset{o_t o_{t+1}}{\mathcal{M}^{-1}}$. In this section, our task is to analyze the rank structure of these tensors and impose restrictions on the sets $\mathbf{O}_L$ and $\mathbf{O}_R$ to ensure that the rank conditions are satisfied. For example, consider equation (2.12) and expand all its terms using (2.10) to get

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times \overbrace{\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times \underbrace{\underset{x_{t-1}d_{t-2}}{\mathcal{K}^{-1}} \times \underset{x_{t-1}d_{t-2}}{\mathcal{K}}} \times \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}} \times$$
$$\times \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}},$$

where we dropped the multiplication subscripts and some of the duplicated modes, which can be inferred from the context. Observe, that in order for the above equation to produce (2.8), the terms in the middle must multiply out into identity tensor

$$\underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{x_{t-1}d_{t-2}}{\mathcal{K}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \qquad \underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}. \tag{2.28}$$

Moreover, recall that $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ was originally introduced as part of the identity tensor

$$\underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{\mathbf{O}_{R_{t-1}}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}, \tag{2.29}$$

therefore, we can conclude that for (2.12) to exist, the identity statements in (2.28) and (2.29) must be satisfied. These statements have implications for the ranks of $\underset{x_{t-1}d_{t-2}}{\mathcal{K}}$, $\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ and $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$, which in turn determine the length of the observation sequences $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$.

Since $\underset{x_{t-1}d_{t-2}}{\mathcal{K}}$ represents a distribution $p(x_{t-1}d_{t-2})$, its matrisized version is a diagonal matrix with $p(x_{t-1}d_{t-2})$ on the diagonal. Using assumptions $A1$ and $A2$, it can be concluded that the diagonal elements in this matrix are non-zero and it has rank $n_x n_d$, it is thus invertible and so the first equation in (2.28) is satisfied.

Next, consider the second equation in (2.28) and recall from Section 2.3.1 that if we matrisize the tensor as $\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathbf{F}} \in \mathbb{R}^{n_o^{|\mathbf{O}_{L_{t-1}}|} \times n_x n_d}$ then $\mathbf{F}$ must have full column rank $n_x n_d$ for the proper inverse to exist, implying $n_o^{|\mathbf{O}_{L_{t-1}}|} \geq n_x n_d$. Similarly, $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}}$ in (2.29) must have rank $n_x n_d$. As a consequence of the above, the tensor

$$\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}} \times \underset{x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{K}}} \tag{2.30}$$

will have rank $n_x n_d$ and, in general, is rank-deficient.

The argument above can also be used to show that $\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\boldsymbol{\mathcal{M}}}$ has rank $n_x n_d$ since in (2.14) the tensors $\underset{x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{K}}}$, $\underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{F}}}$ and $\underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{F}}}$ all have rank $n_x n_d$. Similarly, $\underset{o_t o_{t+1}}{\boldsymbol{\mathcal{M}}}$ will have rank n_x because in (2.17) the rank of the participating tensors $\underset{x_t}{\boldsymbol{\mathcal{K}}}$, $\underset{o_{t+1}|x_t}{\boldsymbol{\mathcal{F}}}$ and $\underset{o_t|x_t}{\boldsymbol{\mathcal{F}}}$ is n_x. In particular, note that the tensor $\underset{o_t|x_t}{\boldsymbol{\mathcal{F}}}$ is the observation matrix $O \in \mathbb{R}^{n_o \times n_x}$ of the model and it has rank n_x according to assumption $A3$. This conclusion also justifies our choice for $\omega_{x_t} = o_t$ at the end of Section 2.3.4.

The key unknowns now are the sets of the observed variables $\mathbf{O}_R$ and $\mathbf{O}_L$ that must be appropriately selected for the corresponding tensors to have rank $n_x n_d$. Recall that we defined $\mathbf{O}_{R_{t-1}} = \{o_t, o_{t+1}, \ldots\}$. As one of the new key results of our work, we established that if we select the observations o_t non-sequentially with gaps that grow exponentially with the state size n_x then the following result holds for all t:

Theorem 1 *Let the number of observations be* $|\mathbf{O}_{R_{t-1}}| = \ell$ *and define the set of indices* $\mathcal{S} = \left\{\max\left[t,\ t + (n_d - 1) - (n_x^i - 1)\right] \mid i = 0, \ldots, \ell - 1\right\}$*, such that* $\mathbf{O}_{R_{t-1}} = \{o_k | k \in \mathcal{S}\}$ *then the rank of tensor* $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\boldsymbol{\mathcal{F}}}$ *is* $\min[n_x^\ell,\ n_x n_d]$*.*

As a consequence of this result, to achieve the rank $n_x n_d$ we will require $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations, since we need to ensure $n_x^\ell \geq n_x n_d$ and we want the minimal ℓ which satisfies this. The span of the selected observations is n_d, while their number is only logarithmic in n_d. For example, consider the estimation of tensor $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\boldsymbol{\mathcal{M}}}$ for an HSMM with $n_x = 3$ and $n_d = 20$. In this case $\ell = 4$ and $\mathbf{O}_{R_{t-1}} = \{o_t, o_{t+11}, o_{t+17}, o_{t+19}\}$ and $\mathbf{O}_{L_{t-1}} = \{o_{t-21}, o_{t-19}, o_{t-13}, o_{t-2}\}$, where the set $\mathbf{O}_{L_{t-1}}$ is defined similar to $\mathbf{O}_{R_{t-1}}$

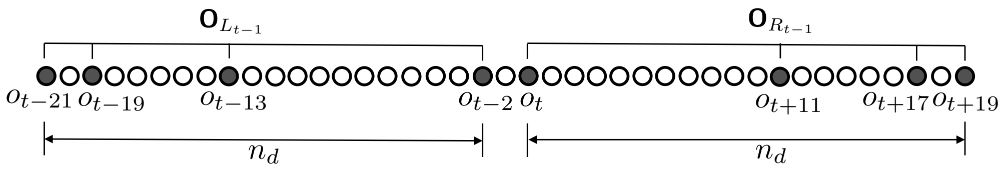


Figure 2.6: Observations required to estimate tensor $\mathcal{M}$ in (2.30) from data for HSMM with $n_x = 3$ and $n_d = 20$.

in Theorem 1 but for the indices to the left of time stamp $t-1$. Figure 2.6 illustrates this example. We note that the requirement for the span of the selected observations to be n_d, which is a maximum state persistence, is to ensure that for a given time stamp t, we select the observations far enough to the right and left of it so that those observations are likely to be sampled from different hidden states.

In order to prove the above Theorem, we will focus our analysis on the tensor $\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathcal{F}}$ instead of $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$. This specific choice was only done to ensure the compactness in our notations, however the HSMM homogeneity property enables us to transfer this result for tensors for any t. Note that

$$\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathcal{F}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-2}d_{t-2}}{\mathcal{F}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathcal{X}}, \tag{2.31}$$

where the first equality is due to the homogeneity property of the model and in the second equality we embedded the HSMM transition matrix into tensor $\underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathcal{X}}$ with mode d_{t-2} duplicated. It can be shown that the matricized tensor $\underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathbf{X}} \in \mathbb{R}^{n_xn_d \times n_xn_d}$ has rank n_xn_d, i.e., it is full rank. Therefore, the rank structure of $\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathcal{F}}$ determines the rank structure of $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$.

The rest of Section 2.4 is devoted to the proof of Theorem 1. We first establish the rank structure of tensor $\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathcal{F}}$ for sequential set of observations $\mathbf{O}_{R_{t+1}}$ and then analyze the rank structure for the observations which were selected non-sequentially.

2.4.1 Rank Structure of Tensor $\mathcal{F}$ in (2.31)

Define by $\mathbf{X}_{R_{t+1}} = \{x_{t+2}, x_{t+3}, \ldots\}$, the sequence of hidden states corresponding to $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \ldots\}$. Then using conditional independence property of the graphical model in Figure 2.1, namely, that the variables $\mathbf{O}_{R_{t+1}}$ and x_td_t are independent given $\boldsymbol{\mathsf{X}}_{R_{t+1}}$, we can write:

$$\underset{\boldsymbol{\mathsf{O}}_{R_{t+1}}|x_t d_t}{\boldsymbol{\mathfrak{F}}} = \underset{\boldsymbol{\mathsf{O}}_{R_{t+1}}|\boldsymbol{\mathsf{X}}_{R_{t+1}}}{\boldsymbol{\mathfrak{Q}}} \times \underset{\boldsymbol{\mathsf{X}}_{R_{t+1}}|x_t d_t}{\boldsymbol{\mathfrak{T}}}, \tag{2.32}$$

for some tensors $\boldsymbol{\mathfrak{Q}}$ and $\boldsymbol{\mathfrak{T}}$, representing the appropriate probability distributions.

Denoting $\ell = |\boldsymbol{\mathsf{O}}_{R_{t+1}}| = |\boldsymbol{\mathsf{X}}_{R_{t+1}}|$, it can be verified, that the matrisized form of $\boldsymbol{\mathfrak{Q}}$ in (2.32) can be written as $\mathbf{Q} = \otimes_\ell O \in \mathbb{R}^{n_o^\ell \times n_x^\ell}$, i.e., a Kronecker product of the observation matrix O with itself ℓ times. According to the assumption $A3$, $rank(O) = n_x$ and $n_x \leq n_o$, and using the rank property of the Kronecker product, we infer that $rank(\mathbf{Q}) = n_x^\ell$.

Combining the above conclusion with the fact that the matrisized form of the other two tensors in (2.32) is $\mathbf{F} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$ and $\mathbf{T} \in \mathbb{R}^{n_x^\ell \times n_x n_d}$, to ensure invertibility of $\boldsymbol{\mathfrak{F}}$, we need to select a set of variables $\boldsymbol{\mathsf{X}}_{R_{t+1}}$ so that $rank\Big(\underset{\boldsymbol{\mathsf{X}}_{R_{t+1}}|x_t d_t}{\mathbf{T}}\Big) = n_x n_d$ with the condition that $n_x^\ell \geq n_x n_d$. Thus, the problem of the analysis of the rank structure of tensor $\underset{\boldsymbol{\mathsf{O}}_{R_{t+1}}|x_t d_t}{\boldsymbol{\mathfrak{F}}}$ translates to the problem of rank structure of matrix

$$\underset{\boldsymbol{\mathsf{X}}_{R_{t+1}}|x_t d_t}{\mathbf{T}} \tag{2.33}$$

In what follows, we assume that $\boldsymbol{\mathsf{X}}_{R_{t+1}} = \{x_{t+2}, \ldots, x_{t+\ell+1}\}$ are sequential and so we would be interested in determining ℓ which makes $rank\Big(\underset{\boldsymbol{\mathsf{X}}_{R_{t+1}}|x_t d_t}{\mathbf{T}}\Big) = n_x n_d$. Later, the sequential assumption will be removed and we show how to select such variables in a more efficient way.

Computation of Factor T in (2.33)

In order to study the rank structure of $\underset{\boldsymbol{\mathsf{X}}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ we will have to understand the mechanism how this matrix is constructed and how the rank changes as the size of $\boldsymbol{\mathsf{X}}_{R_{t+1}}$ increases. We start by considering the following conditional independence relationships from the model in Figure 2.1:

$$p(x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1}) = \sum_{d_{t+2}} p(x_{t+3}|x_{t+2}, d_{t+2}) \underbrace{p(d_{t+2}|x_{t+2}, d_{t+1}) p(x_{t+2}|x_{t+1}, d_{t+1})} \tag{2.34}$$

$$p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t) = \sum_{d_{t+1}} p(x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1}) \underbrace{p(d_{t+1}|x_{t+1}, d_t) p(x_{t+1}|x_t, d_t)}. \tag{2.35}$$

Using the model's homogeneity property, we see that the quantity underlined in (2.34) is the same as the one in (2.35). Moreover, equation (2.34) can then be thought of as transforming $p(x_{t+1}|x_t, d_t)$ into $p(x_{t+2}, x_{t+1}|x_t, d_t)$, while the expression in (2.35) is, in effect, transforms $p(x_{t+2}, x_{t+1}|x_t, d_t)$ into $p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t)$. Thus (2.34) and (2.35) encode the following chain of transformations:

$$p(x_{t+1}|x_t, d_t) \to p(x_{t+2}, x_{t+1}|x_t, d_t) \to p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t).$$

Based on the above considerations, we can rewrite (2.34) and (2.35) in the tensor form as follows:

$$\underset{x_{t+3},x_{t+2}|x_{t+1},d_{t+1}}{\boldsymbol{\mathcal{T}}} = \underset{x_{t+3},x_{t+2}|x_{t+2},d_{t+2}}{\boldsymbol{\mathcal{T}}} \times_{x_{t+2}d_{t+2}} \underset{x_{t+2},d_{t+2}|x_{t+1}d_{t+1}}{\boldsymbol{\mathcal{V}}} \tag{2.36}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\boldsymbol{\mathcal{T}}} = \underset{x_{t+3},x_{t+2},x_{t+1}|x_{t+1},d_{t+1}}{\boldsymbol{\mathcal{T}}} \times_{x_{t+1}d_{d+1}} \underset{x_{t+1},d_{t+1}|x_t d_t}{\boldsymbol{\mathcal{V}}}, \tag{2.37}$$

where $\underset{x_{t+2},d_{t+2}|x_{t+1},d_{t+1}}{\boldsymbol{\mathcal{V}}} = \underset{x_{t+1},d_{t+1}|x_t,d_t}{\boldsymbol{\mathcal{V}}} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\boldsymbol{\mathcal{D}}} \times_{x_{t+1}d_t} \underset{x_{t+1},d_t|x_t,d_t}{\boldsymbol{\mathcal{X}}}$. The homogeneity property allows us to rewrite the above as

$$\underset{x_{t+2},x_{t+1}|x_t,d_t}{\boldsymbol{\mathcal{T}}} = \underset{x_{t+1},x_t|x_t,d_t}{\boldsymbol{\mathcal{T}}} \times \boldsymbol{\mathcal{V}} \tag{2.38}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1},x_{t+1}|x_t,d_t}{\boldsymbol{\mathcal{T}}} = \underset{x_{t+2},x_{t+1}|x_t,d_t}{\boldsymbol{\mathcal{T}}} \times \boldsymbol{\mathcal{V}}. \tag{2.39}$$

Our next step is to represent the above tensor equations in the matrix form. First, consider tensor $\boldsymbol{\mathcal{V}}$, its matricized form can be written as:

$$\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \; \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}} \tag{2.40}$$

where $\underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \in \mathbb{R}^{n_x n_d \times n_x n_d}$ and $\underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}} \in \mathbb{R}^{n_x n_d \times n_x n_d}$. Next, consider the equations (2.38) and (2.39), its matrix version is of the form:

$$\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \; \mathbf{V} \tag{2.41}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \; \mathbf{V}, \tag{2.42}$$

here $\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^2 \times n_x n_d}$, $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^2 \times n_x n_d}$, and similarly $\underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^3 \times n_x n_d}$, and matrix $\underset{x_{t+3},x_{t+2},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^3 \times n_x n_d}$.

Summarizing the above derivations, we can describe the following algorithmic approach for analyzing $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ as $\mathsf{X}_{R_{t+1}}$ increases. We begin with $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}} = [\mathcal{X}\ \mathbf{I}\ \cdots\ \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$, where the first block $\mathcal{X} \in \mathbb{R}^{n_x \times n_x}$ corresponds to $d_t = 1$, and the subsequent $(n_d - 1)$ blocks of $\mathbf{I} \in \mathbb{R}^{n_x \times n_x}$ correspond to $d_t > 1$ for which $x_{t+1} = x_t$. We then use (2.41) to get $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$. However, notice that in (2.41) the matrix $\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}}$ has a duplicated mode x_t, therefore, we need to restructure $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$, which can be accomplished with:

$$\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \mathbf{E},$$

where $\mathbf{E} = [\mathbf{I}\ \cdots\ \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$ and $\odot$ denotes a Khatri-Rao product (row-wise Kronecker product)[1] . Then, we use (2.42) to transform $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$ into $\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$ where, again a preliminary step is to restructure the matrix as follows:

$$\underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \mathbf{E}.$$

Algorithm 3 summarizes the above constructions for a general case. $\underset{X_{R_{t+1}}|x_t d_t}{\mathbf{T}}$

The following Theorem characterizes the rank structure of matrix $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ in the output of the Algorithm 3. The proof can be found in Appendix 2.A.1.

Theorem 2 *The rank of the output matrix* $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ *in Algorithm 3 is* $\min(\ell n_x, n_x n_d)$*.*

Applying now Theorem 2 to equation (2.32) in matrix form

$$\underset{\mathsf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}} = \underset{\mathsf{O}_{R_{t+1}}|\mathsf{X}_{R_{t+1}}}{\mathbf{Q}} \times \underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}},$$

where $rank(\mathbf{Q}) = n_x^\ell$ we can now conclude the following result:

Corollary 3 *To achieve the full column rank for* $\underset{\mathsf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$*, i.e. to ensure that the rank of tensor* $\underset{\mathsf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ *is* $n_x n_d$*, the number of observations* ℓ *in* $\mathsf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \ldots, o_{t+\ell+1}\}$ *must be equal to the maximum state persistence i.e.,* $\ell = n_d$*.*

[1] Let $\mathbf{P} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} \in \mathbb{R}^{m \times n}$ and $\mathbf{Q} \in \mathbb{R}^{k \times n}$ then $\mathbf{P} \odot \mathbf{Q} = \begin{bmatrix} \mathbf{p}_1 \otimes \mathbf{Q} \\ \mathbf{p}_2 \otimes \mathbf{Q} \\ \vdots \\ \mathbf{p}_n \otimes \mathbf{Q} \end{bmatrix} \in \mathbb{R}^{mk \times n}$, where $\otimes$ is a Kronecker product.

Algorithm 3 Computation of $\mathbf{T}$ in (2.33)

```
Input: $p(d_t|x_t, d_{t-1})$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, $\ell$ - the number of sequential hidden states represented by $\mathbf{X}_{R_{t+1}}$.
Initialization:
    $p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$
    $p(d_{t+1}|x_{t+1}, d_t) \rightarrow \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}}$
    $p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}$
    $\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \; \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}, \quad \mathbf{E} = [\mathbf{I} \cdots \mathbf{I}]$
for $i = 1$ to $\ell - 1$ do
    $\underset{x_{t+i},\,\dots\,,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+i},\,\dots\,,x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \mathbf{E}$    (2.43)
    $\underset{x_{t+i+1},\,\dots\,,x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+i},\,\dots\,,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} \; \mathbf{V}$    (2.44)
end for
```

Efficient Computation of Factor T in (2.33)

In Corollary 3 we established that the required number of observations in $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \dots, o_{t+\ell+1}\}$ is $\ell = n_d$. Therefore, the sizes of the estimated quantities $\tilde{\mathfrak{D}} \in \mathbb{R}^{n_o^{n_d} \times n_o^{n_d}}$ and $\tilde{\mathfrak{X}} \in \mathbb{R}^{n_o^{n_d} \times n_o^{n_d} \times n_o}$ in Algorithm 3 will have exponential dependency on n_d. When maximum state persistence is large, the estimation of such quantity becomes impractical. Fortunately, we can modify Algorithm 3 to significantly reduce the number of observations. The idea is to apply the step (2.44) multiple times in-between the applications of step (2.43). Recall that in the previous construction we established that $\ell = n_d$ consecutive observations are sufficient, e.g., $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, \dots, o_{t+\ell+1}\}$. In contrast, in the proposed approach, every time we add an observation, say $o_{t+\tau}$, we skip certain number δ of time steps before adding another observation $o_{t+\tau+\delta}$, so that the observations are non-consecutive. As we illustrate next, the span of these non-consecutive observations is still n_d but the number of them is only logarithmic in

Algorithm 4 Efficient computation of $\mathbf{T}$ in (2.33)

```
Input: $p(d_t|x_t, d_{t-1})$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, $\ell$ - the number of sequential hidden states represented by $\mathsf{X}_{R_{t+1}}$
Initialization:
    $p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$
    $p(d_{t+1}|x_{t+1}, d_t) \rightarrow \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}}$
    $p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}$
    $\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \quad \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}, \quad \mathbf{E} = [\mathbf{I} \cdots \mathbf{I}]$
$c = 1$
for $i = 1$ to $\ell - 1$ do
        $\mathbf{T} = \mathbf{T} \quad \mathbf{V}$        (2.45)
    if $i == (n_x)^c - 1$ or $i == \ell - 1$ do
        $\mathbf{T} = \mathbf{T} \odot \ \mathbf{E}$        (2.46)
    end if
    $c = c + 1$
end for
```

n_d. The proposed approach still achieves the full rank structure of $\underset{\mathsf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}}$ but with smaller number of data points. Algorithm 4, which is a simple modification of Algorithm 3, summarizes the above procedure.

The following result establishes the rank structure of the matrix $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ in the output of the Algorithm 4. The proof can be found in Appendix 2.A.2.

Theorem 4 *The rank of the output matrix* $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ *in Algorithm 4 is* $\min(n_x^\ell, n_x n_d)$.

Note that based on the above theorem, Algorithm 4 increases the rank at every step exponentially rather than linearly. In order for $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ to achieve the rank $n_x n_d$ we will now require $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations, since we need to ensure $n_x^\ell = n_x n_d$. Observe that the span of the selected observations is still n_d, while the number of the observations is only logarithmic in n_d. The following Corollary summarizes the above conclusions.

Corollary 5 *To achieve the full column rank for* $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$*, i.e. to ensure that the rank of tensor* $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathfrak{F}}$ *is* $n_x n_d$*, the number of observations* ℓ *in* $\mathbf{O}_{R_{t+1}}$ *must be equal to* $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$*, since we need to ensure* $n_x^\ell = n_x n_d$*.*

Theorem 4 together with Corollary 5 now proves the Theorem 1 stated earlier.

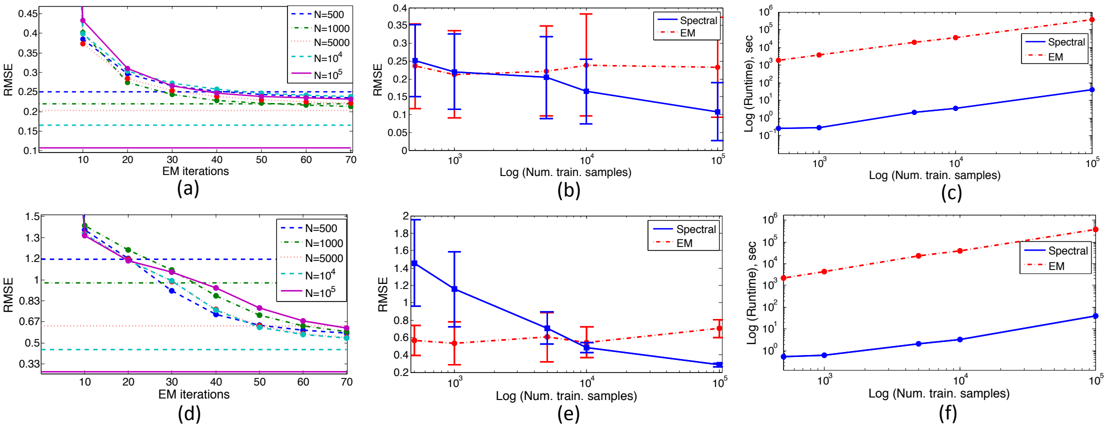
2.5 Experiments

In this section we evaluated the performance of the proposed algorithm both on synthetic as well as real datasets and compared its performance to a standard EM algorithm.

2.5.1 Synthetic Data

Using synthetic data, we compared the estimation accuracy and the runtime of the spectral algorithm with EM. For this, we defined two HSMMs, one with $n_o = 3, n_x = 2, n_d = 2$ and another with $n_o = 5, n_x = 4, n_d = 6$. For each model, we generated a set of $N_{train} = \{500, 1000, 5000, 10^4, 10^5\}$ training and $N_{test} == 1000$ testing sequences, each of length $T = 100$. The accuracy of estimating likelihood for each testing sequence was measured using the relative deviation from the true likelihood, i.e., $\epsilon_i = \frac{|\hat{p}(\mathbf{S}_i^{test}) - p(\mathbf{S}_i^{test})|}{p(\mathbf{S}_i^{test})}$ for $i = 1, \ldots, 1000$. Given 1000 such values, we then computed the final score, which is the root-mean-square error (RMSE) across all the testing sequences, $\text{RMSE} = \sqrt{\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \epsilon_i^2}$.

Figure 5.2 shows results, where the top row of graphs corresponds to the model $n_o = 3, n_x = 2, n_d = 2$ and the bottom row is for model $n_o = 5, n_x = 4, n_d = 6$. The left column of graphs shows the progression of RMSE across EM iterations for both models; the middle column shows the dependence of testing error on the number of training samples and the right column shows the running times. It can be observed from plots (b) and (e) in Figure 5.2 that with the small training set, EM achieves smaller errors,



(a) (b) (c)

(d) (e) (f)

Figure 2.7: Performance of the spectral algorithm and EM on synthetic data generated from HSMM with $n_o = 3, n_x = 2, n_d = 2$ (top row) and $n_o = 5, n_x = 4, n_d = 6$ (bottom row). (a), (d): Error for EM across different iterations for various training datasets. The straight lines show the performance for spectral method. (b), (e): Average error and one standard deviation over 100 runs for EM after convergence and spectral algorithm across different number of training data. (c), (f): Runtime, in seconds, for both methods.

while as the number of training samples increases, the spectral method becomes more accurate, outperforming EM. Also, comparing the plots (a), (b) with (d) and (e), we can conclude that for larger models, i.e., whose n_o, n_x and n_d are larger, the spectral method requires more data in order to achieve same or better accuracy than EM. This is expected since the sizes of estimated tensors grow with the model size. Moreover, the plots (c) and (f) in Figure 5.2 show that spectral method is several orders of magnitude faster than EM.

Given the above results, we can conclude that (i) for small datasets EM is a preferable algorithm, (ii) for large data, the spectral algorithm is a better choice, since it achieves higher accuracy and (iii) across all datasets the spectral algorithm requires significantly less computations as compared to EM.

2.5.2 Application to Aviation Safety Data

We also compared the performance of the spectral algorithm and EM on real NASA flight dataset [42], containing over 180000 flights of 35 aircrafts from a defunct mid-western airline company. For each flight, the data has a record of 186 parameters,

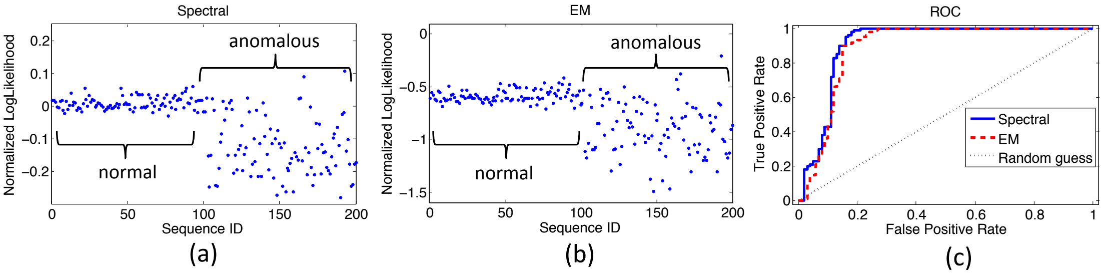


Figure 2.8: Evaluation of the spectral algorithm and EM on aviation safety data. (a) and (b): Normalized joint loglikelihood computed by spectral algorithm (a) and EM (b) for a set of 200 test flights, with 100 normal and 100 anomalous. HSMM parameters: $n_o = 9, n_x = 8, n_d = 40$ (c): The Receiver Operating Characteristic (ROC) curve, illustrating classification accuracy of the algorithms. Area Under Curve (AUC) for spectral algorithm is 0.91 and for EM is 0.89.

sampled at 1 Hz, including sensor readings and pilot actions. We considered a problem of anomaly detection in aviation systems [58, 72, 8] and used HSMM to detect abnormal flights based on pilot actions. Our idea is based on the observation that a flight can be partitioned into a number of phases, e.g., initial descent, touch down, or braking on the runway, and where within each phase the pilot performs certain actions. For example, during the initial descent, the pilot reduces throttle, lowers the flaps, and uses the ailerons and elevator to stabilize the aircraft. On the other hand, in the braking stage, the pilot uses brakes as well as rudder to keep the aircraft in the middle of the runway. Using HSMM as a model, we represented the flight phases as hidden states and the pilot actions as the observations from these states (see [34] for more details).

In our experiments, we focused on a part of flight related to the approach phase (15 − 60 minutes in duration before the touch down on the runway) for a subset of flights landing at the same airport. We chose 9 pilot commands, among which are “selected altitude”, “selected heading”, ”selected throttle level”, etc. A simple data filter, based on the histogram of the pilot actions, was applied to select 10020 normal flights for training. A test set contained 200 flights, with 100 of them being similar to the training set and the rest were selected from the flights rejected by the filter. Most of abnormal flights contained low occurrence events, such as fast descent, unusual usage of air brakes, etc., and few significant anomalies, e.g., the aborted descent in order to delay the flight. The length of the considered sequences varied anywhere from 500 to

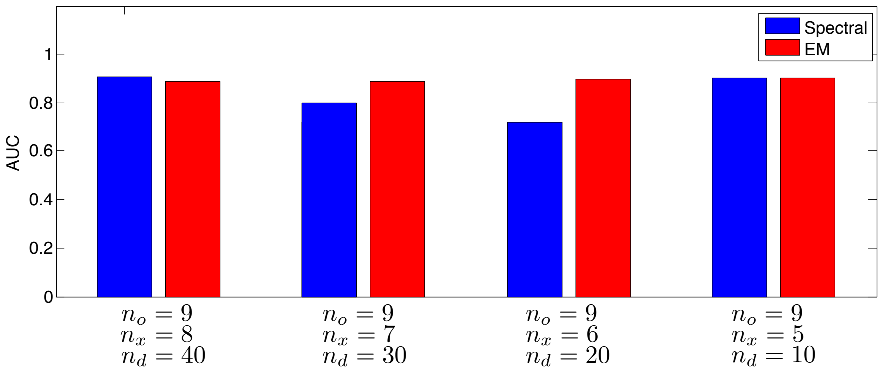


Figure 2.9: Comparison of AUC scores for EM and spectral algorithm for various model parameters when evaluated on aviation safety data. Both algorithms achieve similar high accuracy across different models.

4000 seconds.

Parameters		$n_o = 9$ $n_x = 8$ $n_d = 40$	$n_o = 9$ $n_x = 7$ $n_d = 30$	$n_o = 9$ $n_x = 6$ $n_d = 20$	$n_o = 9$ $n_x = 5$ $n_d = 10$
Running Time	Spectral	6.8 hours	6.4 hours	6.4 hours	6.3 hours
	EM	> 2 days	> 2 days	> 2 days	> 2 days

Table 2.1: Comparison of running time for EM and spectral algorithm for multiple model parameters. Spectral algorithm is several orders of magnitude faster as compared to EM, offering significant computational savings.

We applied EM and spectral algorithm to compute the normalized joint log-likelihood

$$\frac{1}{T_i} \log p(o_1, o_2, \ldots, o_{T_i}),$$

for $i = 1, \ldots, 200$, where o_i are the observed pilot actions. Figure 2.8 shows the results. The high-likelihood sequences were considered normal and low-likelihood ones classified as anomalous (see plots (a) and (b)). Both algorithms achieved similar detection accuracy, with the spectral algorithm having the Area Under Curve (AUC) score of 0.91 and the EM had AUC = 0.89. On the other hand, the computational time of the spectral algorithm was orders of magnitude smaller as compared to EM (see plot (c) on Figure 2.8). We also compared performance of both algorithm on the same flight data while varying the dimensionality of the HSMM parameters (see Figure 2.9 and Table 2.1). We can see that although the performance of EM and spectral algorithm is similar across many models, the latter offers significant computational savings.

Appendix

2.A Analysis of Tensor Rank Structure

2.A.1 Analysis of Algorithm 3

In this Section, we provide analysis of the Algorithm 3 and study the rank structure of matrix $\mathbf{T}$ in order to prove Theorem 2. To understand the analysis, it is important to know how the structure of matrix $\underset{\mathbf{x}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ evolves across iterations. For this, we present in Figure 2.A.1 a schematic description of a few steps of the algorithm. For the analysis we will need to establish certain auxiliary results.

Lemma 6 *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a matrix with no all-zero columns then rank* $(\mathbf{I} \odot \mathbf{A}) =$ *rank* $(\mathbf{A} \odot \mathbf{I}) = n$*, where $\odot$ denotes Khatri-Rao product and $\mathbf{I} \in \mathbb{R}^{n \times n}$.*

Proof Let $\mathbf{K} = (\mathbf{I} \odot \mathbf{A}) \in \mathbb{R}^{mn \times n}$. By definition of Khatri-Rao product, $\mathbf{K}(:, j) = \mathbf{e}_j \otimes \mathbf{A}(:, j)$, for $j = 1, \ldots, n$, which consists of zeros, except for rows $(j-1)m + 1, \ldots, (j-1)m + m$, containing the column $\mathbf{A}(:, j)$. Here $\otimes$ denotes Kronecker product and $\mathbf{e}_j$ is everywhere zero except for position j which is 1. As long as there is no all-zero columns in $\mathbf{A}$, each column of $\mathbf{K}$ is independent of each other and therefore the rank is n. Moreover, since the matrix $\mathbf{A} \odot \mathbf{I}$ is a row-permuted version of $\mathbf{A} \odot \mathbf{I}$, their ranks are the same. ■

Lemma 7 *Define a block-row matrix $\mathbf{M} = [\mathbf{A}_1 \ \mathbf{A}_2 \ \cdots \ \mathbf{A}_k] \in \mathbb{R}^{m \times kn}$, where each $\mathbf{A}_i \in \mathbb{R}^{m \times n}$. Define by r_j, $j = 1, \ldots, n$ the rank of matrix $[\mathbf{A}_1(:, j) \ \cdots \ \mathbf{A}_k(:, j)]$ composed of jth columns of $\mathbf{A}$'s, and let $\mathbf{E} = [\mathbf{I} \ \mathbf{I} \ \cdots \ \mathbf{I}] \in \mathbb{R}^{n \times kn}$, where $\mathbf{I} \in \mathbb{R}^{n \times n}$. Then the rank of matrix $\mathbf{W} = \mathbf{M} \odot \mathbf{E} \in \mathbb{R}^{mn \times kn}$, obtained using a Khatri-Rao product, is* $\min(mn, \sum_j r_j)$*.*

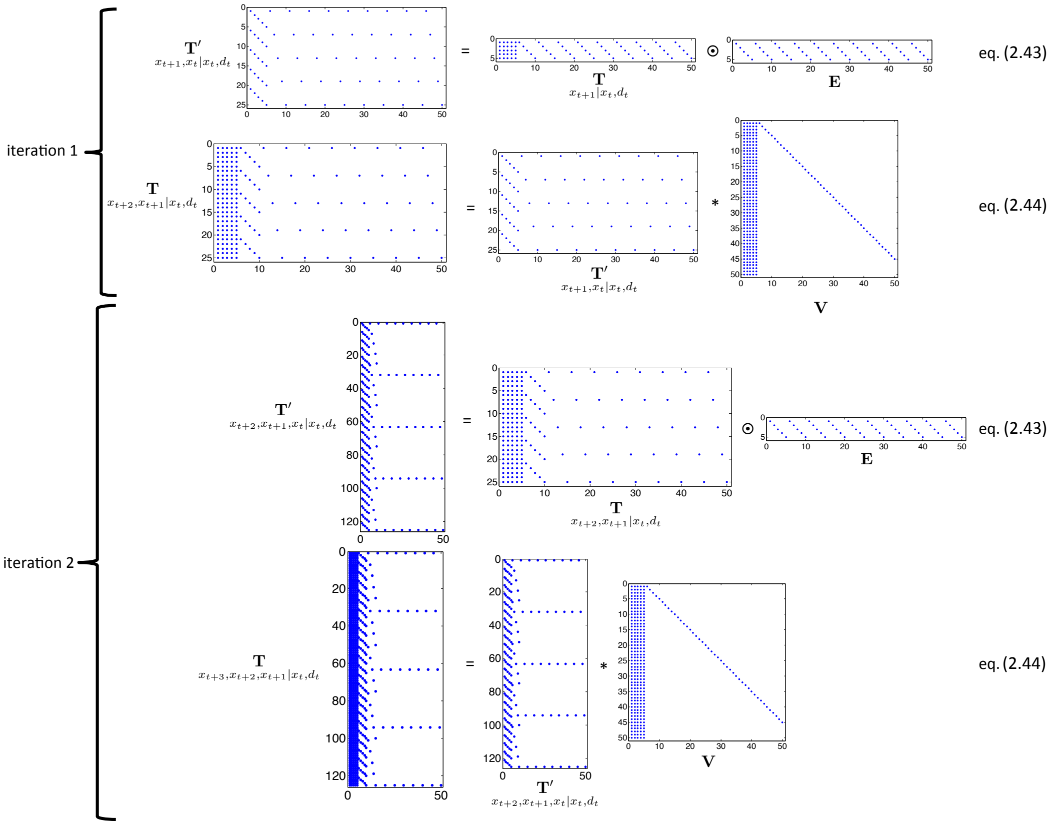

iteration 1
$\mathbf{T}'_{x_{t+1},x_t|x_t,d_t}$
=
$\mathbf{T}_{x_{t+1}|x_t,d_t}$
⊙
$\mathbf{E}$
eq. (2.43)
$\mathbf{T}_{x_{t+2},x_{t+1}|x_t,d_t}$
=
$\mathbf{T}'_{x_{t+1},x_t|x_t,d_t}$
*
$\mathbf{V}$
eq. (2.44)
iteration 2
$\mathbf{T}'_{x_{t+2},x_{t+1},x_t|x_t,d_t}$
=
$\mathbf{T}_{x_{t+2},x_{t+1}|x_t,d_t}$
⊙
$\mathbf{E}$
eq. (2.43)
$\mathbf{T}_{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}$
=
$\mathbf{T}'_{x_{t+2},x_{t+1},x_t|x_t,d_t}$
*
$\mathbf{V}$
eq. (2.44)

Figure 2.A.1: Schematic representation of Algorithm 3. This example illustrates the HSMM with $n_x = 5$ and $n_d = 10$. The non-zero matrix elements are displayed as dots.

Proof First note that $\mathbf{M} \odot \mathbf{E}$ and $\mathbf{E} \odot \mathbf{M}$ are row permuted version of each other, so they have the same rank. Therefore, consider $\mathbf{W}' = \mathbf{E} \odot \mathbf{M} = [\mathbf{I} \odot \mathbf{A}_1 \cdots \mathbf{I} \odot \mathbf{A}_k]$. Also, note that $\mathbf{e}_j \otimes [\mathbf{A}_1(:,j) \cdots \ \mathbf{A}_k(:,j)]$, $j = 1,\ldots,n$ is a matrix which consists of zeros except for rows $(j-1)m+1,\ldots,(j-1)m+m$ where it contains the columns $[\mathbf{A}_1(:,j) \ \cdots \ \mathbf{A}_k(:,j)]$. The rank of these columns is r_j and all other columns in $\mathbf{W}$ are independent of them due to the structure of the Khatri-Rao product. Therefore, each set of such columns adds r_j to the total rank. Since the overall rank of $\mathbf{W}$ cannot exceed either the number of rows or columns, we conclude that $rank(\mathbf{W}) = \min(mn, \sum_j r_j)$. ■

Lemma 8 *Let* $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ *be a set of linearly independent vectors. Define* $\mathbf{u} = \sum_{i=1}^{n} c_i \mathbf{v}_i$*, where coefficients* $c_i \neq 0, i = 1, \ldots, n$*. Define* U *to be a strict subset of* V*, i.e.,* $U \subset V$*, then a set of vectors* $\mathbf{u} \cup U$ *is independent.*

Proof Define $\{1, \ldots, n\} = \alpha \cup \bar{\alpha}$, where α denotes a subset of indices for vectors corresponding to U. Then we can write $\mathbf{u} = \sum_{i:i\in\alpha} c_i \mathbf{v}_i + \sum_{j:j\in\bar{\alpha}} c_j \mathbf{v}_j$.

Assuming the opposite, i.e., $\mathbf{u} \cup U$ are dependent, we can write $k_0 \mathbf{u} + \sum_{i:i\in\alpha} k_i \mathbf{v}_i = 0$ where $k_0 \neq 0$ and some of $k_i, i \in \alpha$ are also must be non-zero. Substituting the definition of $\mathbf{u}$ and rearranging the terms, we get:

$$k_0 \sum_{i:i\in\alpha} (c_i + k_i)\mathbf{v}_i + k_0 \sum_{j:j\in\bar{\alpha}} c_j \mathbf{v}_j = 0.$$

Since $c_j \neq 0, j \in \bar{\alpha}$, the above equation claims the linear dependence of vectors in V, which is a contradiction of our assumption and so $\mathbf{u} \cup U$ are independent. ■

We are now ready to analyze Algorithm 3. It can be verified that (2.40) is of the form:

$$\mathbf{V} = \left[\begin{array}{c|ccc} & \mathbf{I} & & \\ \Psi & & \ddots & \\ & & & \mathbf{I} \\ \hline & \mathbf{0} & \cdots & \mathbf{0} \end{array}\right] \in \mathbb{R}^{n_x n_d \times n_x n_d} \quad \text{where} \quad \Psi = \begin{bmatrix} \operatorname{diag}[D(1,:)]\,\mathcal{X} \\ \operatorname{diag}[D(2,:)]\,\mathcal{X} \\ \vdots \\ \operatorname{diag}[D(n_d,:)]\,\mathcal{X} \end{bmatrix} \in \mathbb{R}^{n_x n_d \times n_x}, \tag{2.47}$$

where $\operatorname{diag}[D(i,:)]$ is the diagonal matrix with ith row from D on the diagonal. Note that we can also write $\Psi = (D \odot \mathbf{I})\,\mathcal{X}$. Observe that the rank of $\mathbf{V}$ is $n_x n_d$ because the $n_x(n_d - 1) \times n_x(n_d - 1)$ block diagonal matrix delineated in (2.47) and the last $n_x \times n_x$ block matrix $\operatorname{diag}[D(n_d,:)]\,\mathcal{X}$ in Ψ together comprising $n_x n_d$ independent columns of $\mathbf{V}$. Note that $\operatorname{diag}[D(n_d,:)]\,\mathcal{X}$ has rank n_x because $\mathcal{X}$ is full rank and $D(n_d,:)$ is non-zero, which follows from assumptions $A1$ and $A2$. As a side note observe that the requirement to have $D(n_d,:)$ non-zero implies that there is a non-zero probability of maximum state persistence.

In analyzing the Algorithm 3, it would be useful to denote the matrices at iteration i in (2.43) and (2.44) as

$$
\begin{aligned}
\underset{x_{t+i},\ \dots\ ,x_{t+1}|x_t,d_t}{\mathbf{T}} &= [\mathbf{A}_1^{(i)}\ \cdots\ \mathbf{A}_{n_d}^{(i)}] \\
\underset{x_{t+i},\ \dots\ ,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} &= [\mathbf{B}_1^{(i)}\ \cdots\ \mathbf{B}_{n_d}^{(i)}] \\
\underset{x_{t+i+1},\dots,x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} &= [\mathbf{C}_1^{(i)}\ \cdots\ \mathbf{C}_{n_d}^{(i)}].
\end{aligned}
$$

Moreover, utilizing the structure of matrix $\mathbf{V}$ from (2.47), the operations involved in step (2.44) are as follows:

$$
\Big[\mathbf{C}_1^{(i)}\ \ \mathbf{C}_2^{(i)}\ \ \mathbf{C}_3^{(i)}\ \ \ \cdots\ \mathbf{C}_{n_d}^{(i)}\Big] = \Big[[\mathbf{B}_1^{(i)}\ \cdots\ \mathbf{B}_{n_d}^{(i)}]\Psi\ \ \ \mathbf{B}_1^{(i)}\ \ \mathbf{B}_2^{(i)}\ \ \ \cdots\ \ \mathbf{B}_{n_d-1}^{(i)}\Big]. \tag{2.48}
$$

With the above information we can now present the proof of Theorem 2:

Proof of Theorem 2 At the start of the algorithm, we have $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}} = [\mathcal{X}\ \mathbf{I}\ \cdots\ \mathbf{I}] = [\mathbf{A}_1^{(1)} \cdots \mathbf{A}_{n_d}^{(1)}]$, which has rank n_x. The rank of matrix $\Big[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\Big]$ for $l = 1,\dots,n_x$ is $r_l = 2$ since among all the columns only two of them are independent. Therefore, according to Lemma 7, the result of operations in (2.43), has rank $\sum_l r_l = 2n_x$. Moreover, we note that since $[\mathbf{B}_1^{(1)}\ \mathbf{B}_2^{(1)}\ \cdots\ \mathbf{B}_{n_d}^{(1)}] = [\mathcal{X}\odot\mathbf{I}\ \ \mathbf{I}\odot\mathbf{I}\ \cdots\ \mathbf{I}\odot\mathbf{I}]$, it can be seen that its $2n_x$ independent vectors can be formed by the columns $[\mathbf{B}_1^{(1)}\ \mathbf{B}_2^{(1)}]$, so that the rank of $\Big[\mathbf{B}_1^{(1)}(:,l) \cdots \mathbf{B}_{n_d}^{(1)}(:,l)\Big]$ for $l = 1,\dots,n_x$ is 2.

Next, since the rank of $\mathbf{V}$ is $n_x n_d$, the operations in (2.44) produce the matrix $[\mathbf{C}_1^{(1)}\ \mathbf{C}_2^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}]$ with the rank still being $2n_x$. Moreover, the columns of $\mathbf{C}_1^{(1)}$ are linearly dependent on the rest of the columns, $[\mathbf{C}_2^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}]$, due to (2.48). However, the rank of $\Big[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\Big]$ is now $r_l = 3$ for $l = 1,\dots,n_x$. To understand this, note that

$$
\begin{aligned}
[\mathbf{B}_1^{(1)}\ \ \mathbf{B}_2^{(1)}\ \ \cdots\ \mathbf{B}_{n_d}^{1}] &= [\mathcal{X}\odot\mathbf{I}\ \ \mathbf{I}\odot\mathbf{I}\ \cdots\ \mathbf{I}\odot\mathbf{I}] \\
[\mathbf{C}_1^{(1)}\ \ \mathbf{C}_2^{(1)}\ \ \mathbf{C}_3^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}] &= [\mathbf{C}_1^{(1)}\ \ \mathcal{X}\odot\mathbf{I}\ \ \mathbf{I}\odot\mathbf{I}\ \cdots\ \mathbf{I}\odot\mathbf{I}],
\end{aligned}
$$

where, according to (2.48), $\mathbf{C}_1^{(1)} = [\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}]\Psi$. As we established before, the rank of the matrix $\Big[\mathbf{C}_2^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\Big] = \Big[\mathbf{B}_1^{(1)}(:,l) \cdots \mathbf{B}_{n_d-1}^{(1)}(:,l)\Big]$ is $r_l = 2$. Moreover, it can also be checked that $\mathbf{C}_1^{(1)}(:,l)$ is independent of $\Big[\mathbf{C}_2^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\Big]$ due to Lemma 8. Clearly, then the cumulative rank of $\Big[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\Big]$ is 3 for $l = 1,\dots,n_x$.

To generalize, if at the iteration i the rank of $\left[\mathbf{A}_1^{(i)} \cdots \mathbf{A}_{n_d}^{(i)}\right]$ is in_x while the rank of $\left[\mathbf{A}_1^{(i)}(:,l) \cdots \mathbf{A}_{n_d}^{(i)}(:,l)\right]$ is $(i+1)$, then the operations in step (2.43) produce $\left[\mathbf{B}_1^{(i)} \cdots \mathbf{B}_{n_d}^{(i)}\right]$ having rank $(i+1)n_x$ due to Lemma 7. The step in (2.44) keeps the rank of $\left[\mathbf{C}_1^{(i)} \cdots \mathbf{C}_{n_d}^{(i)}\right]$ at $(i+1)n_x$ due to the full rank structure of $\mathbf{V}$. At the same time, this step increases the rank of $\left[\mathbf{C}_1^{(i)}(:,l) \cdots \mathbf{C}_{n_d}^{(i)}(:,l)\right]$ to $(i+2)$ due to Lemma 8, i.e., independence of $\mathbf{C}_1^{(i)}(:,l)$ from $\left[\mathbf{C}_2^{(i)}(:,l) \cdots \mathbf{C}_{n_d}^{(i)}(:,l)\right]$ with the latter having the rank $(i+1)$. Therefore, each iteration increases the rank of matrix $\mathbf{T}$ by n_x and so after $2 \leq \ell \leq n_d$ steps the rank of the resulting matrix $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ is ℓn_x.

Note that if $\ell = 1$ then the Algorithm 3 is not executed and returns the trivial $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$ with rank n_x. On the other hand, if $\ell > n_d$ then the rank of $\underset{\mathsf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ is $n_x n_d$ since this is the number of columns in that matrix and so is the maximum achievable rank. ■

2.A.2 Analysis of Algorithm 4

In this Section we analysis of the Algorithm 4 in order to prove Theorem 4. Similarly as in Section 2.A.1, it is instructive to visualize the progress of Algorithm 4. Figure 2.A.2 shows a schematic description of a few steps of the algorithm.

We are now ready to present the proof of Theorem 4.

Proof of Theorem 4 For the proof, we refer back to Algorithm 3 and the proof of Theorem 2. Recall, that at iteration $i = 1$, the result of step (2.43) is a matrix $[\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}] \in \mathbb{R}^{n_x^2 \times n_x n_d}$, whose rank is $2n_x$, since $\left[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\right] = [\mathcal{X}\ \mathbf{I} \cdots \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$ for $l = 1, \dots, n_x$ had two independent columns. Then, the transformations in step (2.44) produced $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ for $l = 1, \dots, n_x$ with rank $3n_x$.

Note that if $n_x > 2$ then $\left[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\right]$ potentially can have a rank up to n_x, while in Algorithm 3 we only have it equal to 2. It turns out that if we apply step (2.44) multiple times and use Lemma 8, we can increase the rank of $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ for $l = 1, \dots, n_x$ to n_x.

Specifically, consider the step (2.45). Then at iteration $i = 1$ we have $[\mathbf{A}_1^{(1)} \cdots \mathbf{A}_{n_d}^{(1)}] =$

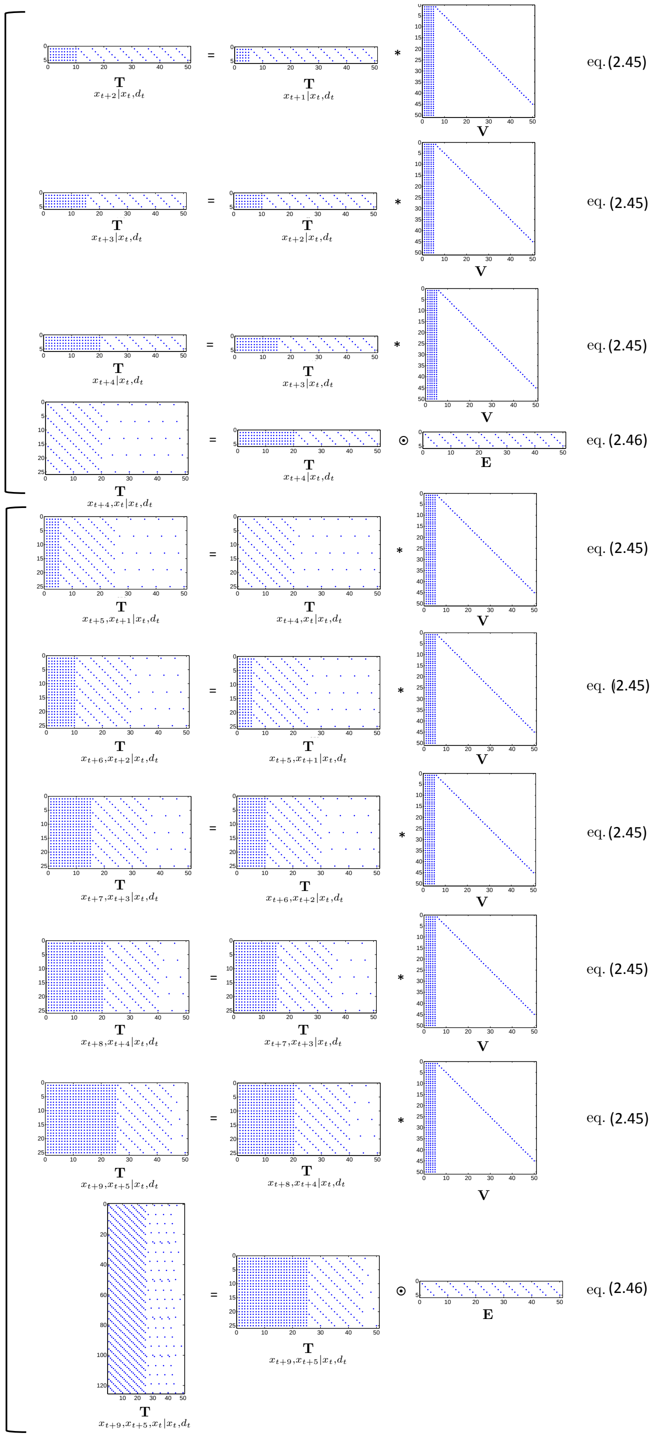

$\mathbf{T}_{x_{t+2}|x_t,d_t}$ = $\mathbf{T}_{x_{t+1}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+3}|x_t,d_t}$ = $\mathbf{T}_{x_{t+2}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+4}|x_t,d_t}$ = $\mathbf{T}_{x_{t+3}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+4},x_t|x_t,d_t}$ = $\mathbf{T}_{x_{t+4}|x_t,d_t}$ $\odot$ $\mathbf{E}$ eq. (2.46)
$\mathbf{T}_{x_{t+5},x_{t+1}|x_t,d_t}$ = $\mathbf{T}_{x_{t+4},x_t|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+6},x_{t+2}|x_t,d_t}$ = $\mathbf{T}_{x_{t+5},x_{t+1}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+7},x_{t+3}|x_t,d_t}$ = $\mathbf{T}_{x_{t+6},x_{t+2}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+8},x_{t+4}|x_t,d_t}$ = $\mathbf{T}_{x_{t+7},x_{t+3}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+9},x_{t+5}|x_t,d_t}$ = $\mathbf{T}_{x_{t+8},x_{t+4}|x_t,d_t}$ * $\mathbf{V}$ eq. (2.45)
$\mathbf{T}_{x_{t+9},x_{t+5},x_t|x_t,d_t}$ = $\mathbf{T}_{x_{t+9},x_{t+5}|x_t,d_t}$ $\odot$ $\mathbf{E}$ eq. (2.46)
0 10 20 30 40 50
0 5 10 15 20 25 30 35 40 45 50
0 20 40 60 80 100 120

Figure 2.A.2: Schematic representation of Algorithm 4. This example illustrates the HSMM with $n_x = 5$ and $n_d = 10$. The non-zero matrix elements are displayed as dots.

$[\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}]$ and for $l = 1, \ldots, n_x$ the two independent columns are $\left[\mathbf{B}_1^{(1)}(:,l) \;\; \mathbf{B}_2^{(1)}(:,l)\right] = [\mathcal{X}(:,l) \;\; \mathbf{I}(:,l)]$. The result of step (2.45) gives us then three independent columns

$$\left[\mathbf{C}_1^{(1)}(:,l) \;\; \mathbf{C}_2^{(1)}(:,l) \;\; \mathbf{C}_3^{(1)}(:,l)\right] = \left[\mathbf{C}_1^{(1)}(:,l) \;\; \mathcal{X}(:,l) \;\; \mathbf{I}(:,l)\right],$$

where $\mathbf{C}_1^{(1)} = [\mathcal{X} \; \mathbf{I} \; \cdots \; \mathbf{I}]\Psi$. The independence follows from Lemma 8. The repeated application of step (2.45) one more time gives four independent columns

$$\left[\mathbf{C}_1^{(2)}(:,l) \;\; \mathbf{C}_2^{(2)}(:,l) \;\; \mathbf{C}_3^{(2)}(:,l) \;\; \mathbf{C}_4^{(2)}(:,l)\right] = \left[\mathbf{C}_1^{(2)}(:,l) \;\; \mathbf{C}_1^{(1)}(:,l) \;\; \mathcal{X}(:,l) \;\; \mathbf{I}(:,l)\right],$$

where $\mathbf{C}_1^{(2)} = [\mathbf{C}_1^{(1)} \cdots \mathbf{C}_{n_d}^{(1)}]\Psi$. Observe that since the number of rows is n_x, we can increase the rank at most up to n_x. Therefore, if in the beginning we had *two* independent columns and we want to get n_x independent columns, we would need to apply the step (2.45) $n_x - 2$ times, so as to have the matrix $[\mathbf{C}_1^{(n_x-2)}(:,l) \; \cdots \; \mathbf{C}_{n_d}^{(n_x-2)}(:,l)]$ with rank n_x.

If we now apply step (2.46) it will give us $[\mathbf{A}_1^{(1)} \; \cdots \; \mathbf{A}_{n_d}^{(1)}] \in \mathbb{R}^{n_x^2 \times n_x n_d}$ with rank n_x^2 due to Lemma 7. Continuing in this manner, we can again repeatedly apply the step (2.45) to create a matrix with a rank at most n_x^2, since there are n_x^2 rows and assuming that $n_x n_d \geq n_x^2$. The number of times we need to apply (2.45) is now $n_x^2 - n_x$ since we need to go from n_x to n_x^2 independent columns.

In general, the step (2.45) needs to be applied $n_x^c - n_x^{c-1}$, in order to obtain n_x^c independent columns. The application of step (2.46) then creates $\mathbf{T}$ with rank n_x^{c+1}. Note, that since $\mathbf{T}$ has $n_x n_d$ columns, the maximum achievable rank is $n_x n_d$. ■

Observe that the above proof also provided the method for selecting the non-sequential observations $\mathbf{X}_{R_{t+1}}$. Specifically, since the set of observations $\mathbf{X}_{R_{t+1}} = \{o_{t+2}, \ldots\}$ must start from observation o_{t+2} and $|\mathbf{X}_{R_{t+1}}| = \ell$, we denote $s = t+2$. Then, ith added observation is $o_{s+(n_d-1)-(n_x^i-1)}$ for $i = 0, \ldots, \ell - 2$ and the ℓth observation is $o_s = o_{t+2}$. For tensor $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ to achieve rank $n_x n_d$ we need to add $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

2.B Initial and Final Parts of HSMM

In this Section we present the derivations for the initial and final steps of HSMM, which were omitted from the main text. Specifically, this amounts to computing the factor $\mathcal{X}$ for two parts of the model, corresponding to $\mathbb{X}_{root}$ and $\mathbb{X}_T$ in Figures 2.B.1 and 2.B.2. The derivations for all other parts of HSMM were presented in the main text and this supplement.

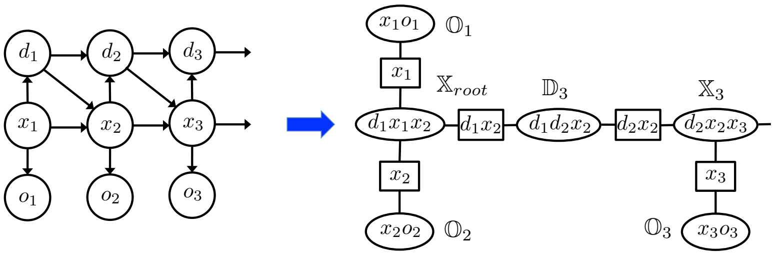


Figure 2.B.1: Part of HSMM corresponding to the initial time stamps and the related part of junction tree.

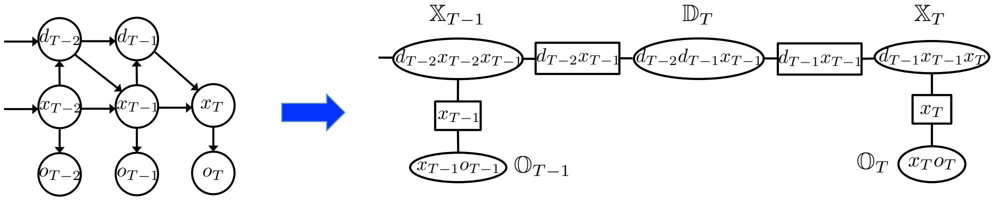


Figure 2.B.2: Part of HSMM corresponding to the final time stamps and the related part of junction tree.

To begin, recall the expression for the joint likelihood of the observed sequence:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \prod_t \underset{d_{t-1}|x_{t-1}d_{t-2}}{\mathfrak{D}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t|x_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathfrak{O}} \right)$$

and rewrite the above expression by keeping only the initial and final factors:

$$\underset{o_1,\ldots,o_T}{\boldsymbol{\mathcal{P}}} = \left(\underset{o_1|x_1}{\boldsymbol{\mathcal{O}}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\boldsymbol{\mathcal{X}}} \times_{x_2} \underset{o_2|x_2}{\boldsymbol{\mathcal{O}}}\right)\right) \times_{x_2d_1} \underset{d_2|x_2x_2d_1}{\boldsymbol{\mathcal{D}}} \times \cdots$$

$$\cdots \times \underset{d_{T-1}|x_{T-1}x_{T-1}d_{T-2}}{\boldsymbol{\mathcal{D}}} \times_{x_{T-1}d_{T-1}} \left(\underset{x_T|x_{T-1}d_{T-1}}{\boldsymbol{\mathcal{X}}} \times_{x_T} \underset{o_T|x_T}{\boldsymbol{\mathcal{O}}}\right). \tag{2.49}$$

Introduce the identity tensors into (2.49), regroup the terms and extract the factors $\boldsymbol{\mathcal{X}}$:

$$\underset{\omega_{x_1}\omega_{x_2}\omega_{x_2d_1}}{\tilde{\boldsymbol{\mathcal{X}}}} = \underset{\omega_{x_1}|x_1}{\boldsymbol{\mathcal{F}}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\boldsymbol{\mathcal{X}}} \times_{x_2} \underset{\omega_{x_2}|x_2}{\boldsymbol{\mathcal{F}}}\right) \times_{x_2d_1} \underset{\omega_{x_2d_1}|x_2d_1}{\boldsymbol{\mathcal{F}}} \tag{2.50}$$

$$\underset{\omega_{x_{T-1}d_{T-1}}\omega_{x_T}}{\tilde{\boldsymbol{\mathcal{X}}}} = \underset{\omega_{x_{T-1}d_{T-1}}|x_{T-1}d_{T-1}}{\boldsymbol{\mathcal{F}}^{-1}} \times_{x_{T-1}d_{T-1}} \left(\underset{x_T|x_{T-1}d_{T-1}}{\boldsymbol{\mathcal{X}}} \times_{x_T} \underset{\omega_{x_T}|x_T}{\boldsymbol{\mathcal{F}}}\right). \tag{2.51}$$

Defining the observable sets $\omega_{x_1} = o_1$, $\omega_{x_2} = o_2$ and $\omega_{x_2d_1} = \mathbf{O}_{R_3}$ we can rewrite (2.50) as follows:

$$\underset{o_1o_2\mathbf{O}_{R_3}}{\tilde{\boldsymbol{\mathcal{X}}}} = \underset{o_1|x_1}{\boldsymbol{\mathcal{F}}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\boldsymbol{\mathcal{X}}} \times_{x_2} \underset{o_2|x_2}{\boldsymbol{\mathcal{F}}}\right) \times_{x_2d_1} \underset{\mathbf{O}_{R_3}|x_2d_1}{\boldsymbol{\mathcal{F}}}. \tag{2.52}$$

Note that since all the factors participating in (2.52) are valid probability distributions, the resulting factor, i.e., $\underset{o_1o_2\mathbf{O}_{R_3}}{\tilde{\boldsymbol{\mathcal{X}}}}$ is also a valid probability distribution, so it can be estimated directly from data. This is in contrast to the derivations we made for other parts of the model, where we had to perform additional transformations such as, for example in (2.12), in order to bring to the form, which could be estimated from the data samples.

In order to estimate (2.51), we compare it to the similar factor we considered in the main paper:

$$\underset{\omega_{x_{t-1}d_{t-1}}\omega_{x_t}\omega_{x_td_{t-1}}}{\tilde{\boldsymbol{\mathcal{X}}}} = \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{F}}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}x_{t-1}d_{t-1}}{\boldsymbol{\mathcal{X}}} \times_{x_t} \underset{\omega_{x_t}|x_t}{\boldsymbol{\mathcal{F}}}\right) \times_{x_td_{t-1}} \underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\boldsymbol{\mathcal{F}}}, \tag{2.53}$$

and observe that the last factor $\underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\boldsymbol{\mathcal{F}}}$ in (2.53) is a conditional probability distribution, which has the following marginalization property

$$\underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\boldsymbol{\mathcal{F}}} \times_{\omega_{x_td_{t-1}}} \underset{\omega_{x_td_{t-1}}}{\mathbf{1}} = \underset{x_td_{t-1}}{\mathbf{1}}, \tag{2.54}$$

where $\mathbf{1}$ is the tensor, which has all elements equal to 1. The above can also be written in the scalar notations, $\sum_{\omega_{x_td_{t-1}}} p(\omega_{x_td_{t-1}}|x_td_{t-1}) = 1$ for each value of x_td_{t-1}. Therefore, if we apply (2.54) to (2.53), we get $\underset{\omega_{x_{t-1}} d_{t-1} \omega_{x_t}}{\tilde{\mathcal{X}}}$, which is the time-shifted version of $\underset{\omega_{x_{T-1}} d_{T-1} \omega_{x_T}}{\tilde{\mathcal{X}}}$. Therefore, to compute (2.51), we estimate the tensor in (2.15), i.e.,

$$\underset{\mathbf{o}_{R_t} o_t \mathbf{o}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{o}_{L_t} \mathbf{o}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{o}_{L_t}} \underset{\mathbf{o}_{L_t} \mathbf{o}_{R_t} o_t}{\mathcal{M}},$$

and marginalize out the right set of modes, corresponding to $\mathbf{o}_{R_t}$. Alternatively, we can use the batch estimate

$$\tilde{\mathcal{X}} = \left(\sum_t \underset{\mathbf{o}_{L_t} \mathbf{o}_{R_t}}{\mathcal{M}} \right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_t} \mathbf{o}_{R_t} o_t}{\mathcal{M}} \right),$$

and similarly perform the marginalization. This concludes our derivations. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Chapter 3

Vector Autoregressive Model: Continuous Data Modeling

In this chapter, our objective is to develop a framework to identify operationally significant events in the flight data, composed from multivariate continuous time series. We are specifically interested in the scenarios when no information about the labels of the flights is available (i.e., which flights are normal and anomalous). To detect the abnormal flights, it is necessary to define a distance measure to compare the flights. And since the flights are represented as multivariate time series, possibly with different lengths, it is unclear how to compare such data objects.

3.1 Introduction

For the continuous data, inspired by viewing the flight data in Figure 1.2 as coming from a dynamical system with certain inputs (corresponding to environment and some control variables) and outputs (corresponding to sensor measurements) and by utilizing ideas from the system identification literature [50], we propose to represent such data as a vector autoregressive model (VAR) (note that the VAR model with exogenous (input) variables is also referred as VARX). Note that the VARX modeling enables us to exploit the relationships between the data parameters and compare the flights with different durations. Moreover, our approach allows online anomaly detection, i.e., analyzing data as it is coming in and compute anomaly score for each time stamp.

We note that in literature, the problem of anomaly detection based on continuous data was addressed by several researchers. In particular, [73] considered a problem of detecting abnormal fuel consumption in jet engines. Their method is based on using regression models to estimate the consumed fuel and compare it to the actual recorded level to detect the abnormal behavior. The proposed method is a supervised approach in which model training requires anomaly-free data, which might limit its practical application in cases when the labeled data is unavailable, as is the case in the present work.

In the work of [72] the authors addressed a general aircraft anomaly detection problem. Their approach is based on using a specially designed linear regression model to describe the aerodynamic forces acting on an aircraft. The constructed model accounts for the flight-to-flight and aircraft-to-aircraft variability, which enables the fitting of a single model to the entire dataset. Hotelling T2 statistics is then used on the residuals to detect anomalies. However, the postulated aerodynamics regression model requires significant domain knowledge and careful design, limiting its generalization and usage in other anomaly detection problems. On the other hand, the current VARX-based approach requires only basic knowledge about the considered parameters to define a model and can easily be extended to other anomaly detection domains.

Das et. al. [74] proposed multiple kernel learning approach for heterogeneous anomaly detection problems (MKAD). The method constructs a kernel matrix as a convex combination of a kernel over discrete sequences using normalized longest common subsequence [58] and a kernel based on symbolic aggregate approximation (SAX) representation [75] of the continuous time series. One-class SVM [76] is then used to construct a separating hyperplane to detect anomalies. This method was applied to the FOQA dataset [8] and showed high accuracy in discovering operationally significant aviation safety events.

3.2 Description of Anomaly Detection Approach

Our approach for analyzing multivariate time series utilizes ideas from system identification [50] and model-based sequential data clustering [77], [78]. In particular, we represent each flight with a Vector AutoRegressive eXogenous model (VARX) [79], [80], which can capture the dependencies among different time series over time. To avoid

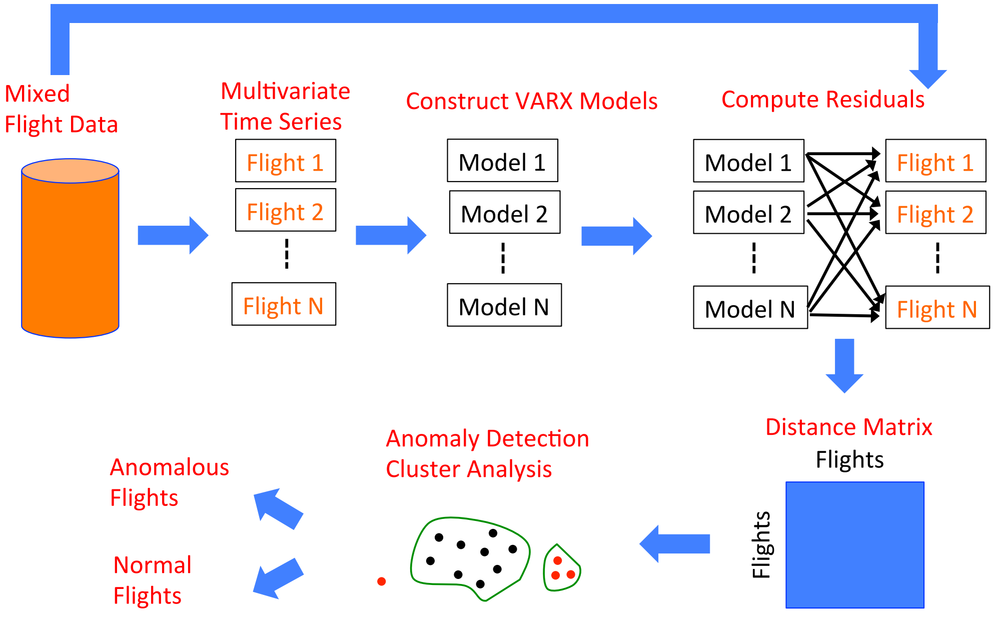


Figure 3.1: Anomaly detection framework using VARX modeling.

deviations due to data noise and outliers, we focus on a robust VARX model, which considers a robust Huber loss [1] instead of a square loss, and develop an efficient method for estimating model parameters based on iterative re-weighted least squares. A distance between flights is defined in terms of residuals of modeling one flight's data using another flight's VARX model, with suitable normalization and symmetrization. The flight-by-flight distance matrix can then be used in any nearest-neighbor based anomaly detection method [81]. Flowchart of the proposed framework is shown in Figure 3.1.

It is important to emphasize a few key aspects of our framework: (i) the VARX modeling enables us to exploit the relationships between the data parameters and compare the flights with different durations, (ii) our approach allows online anomaly detection, i.e., analyzing data as it is coming in, in contrast to dynamic programming based methods such as DTW or LCS (longest common subsequence) [82], which need the entire time series for analysis, and (iii) the framework is scalable, due to the inherent parallel nature of most computations.

In what follows we present the details of the proposed framework and for this purpose, we discuss in details each module from the flowchart digram in Figure 3.1.

3.2.1 Model Construction

We propose to view the aviation data as coming from a dynamical system with certain inputs and outputs. A standard approach for modeling such system in the system identification literature [50] is a vector autoregressive model with exogenous variables (VARX):

$$y_k = A_1 y_{k-1} + \ldots + A_p y_{k-p} + B_1 u_{k-1} + \ldots + B_q u_{k-q} + \epsilon_k, \tag{3.1}$$

where $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$ are the matrices of coefficients, $y \in \mathbb{R}^n$ is the vector of outputs corresponding to the sensor measurements on the aircraft, $u \in \mathbb{R}^m$ is the vector of inputs corresponding to environmental and control features, $\epsilon \in \mathbb{R}^n$ is the vector of zero-mean white noise, and $k = \max(p+1, q+1), \ldots, T$, where T denotes the length of time series. The subscripts p and q determine the lag for y and u, respectively. For future references, we denote the data of each flight, consisting of inputs and outputs, in the dataset of N flights as F^i, so that $F^i_k = \left\{ y^i_k, u^i_k \right\}$, where $k = \max(p+1, q+1), \ldots, T_i$ and $i = 1, \ldots, N$.

Without the loss of generality and to simplify the notations, in the following we consider a first order VARX model

$$y_k = A y_{k-1} + B u_{k-1} + \epsilon_k. \tag{3.2}$$

The key step of our anomaly detection framework is the estimation of such a model for each flight, which amounts to computing the coefficient matrices A and B, and in what follows, we discuss the procedure for estimation of these parameters. For this purpose, we assume that the length of some flight is T timestamps, then it follows that $k = 2, \ldots, T$ (since we consider a first order VARX model) and we can write the expression in (3.2) in the following form

$$\begin{bmatrix} y_2 \\ \vdots \\ y_T \end{bmatrix} = A \begin{bmatrix} y_1 \\ \vdots \\ y_{T-1} \end{bmatrix} + B \begin{bmatrix} u_1 \\ \vdots \\ u_{T-1} \end{bmatrix} + \begin{bmatrix} \epsilon_2 \\ \vdots \\ \epsilon_T \end{bmatrix},$$

which can also be compactly written as

$$Y_{2:T} = AY_{1:T-1} + BU + E,$$
$$Y_{2:T} = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} Y_{1:T-1} \\ U \end{bmatrix} + E.$$

Next, applying the vectorization operation to the above, we get

$$vec(Y_{2:T}) = \begin{bmatrix} Y_{1:T-1}^T \otimes I & U^T \otimes I \end{bmatrix} \begin{bmatrix} vec(A) \\ vec(B) \end{bmatrix} + vec(E)$$
$$\mathbf{y} = Z\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{3.3}$$

where $vec(\cdot)$ is matrix vectorization, i.e., the operation of stacking the columns of a matrix into a vector, $\mathbf{y} \in \mathbb{R}^{n(T-1)}$, $Z = [Y_{1:T-1}^T \otimes I \quad U^T \otimes I] \in \mathbb{R}^{n(T-1)\times(n^2+nm)}$, $\boldsymbol{\beta} = \begin{bmatrix} vec(A) \\ vec(B) \end{bmatrix} \in \mathbb{R}^{(n^2+nm)}$ and $I \in \mathbb{R}^{n\times n}$ is the identity matrix. The symbol $\otimes$ denotes the Kronecker product operation [83]. In the above we used a fact that $vec(PQ) = (Q^T \otimes I)vec(P)$, for matrices with appropriate dimensions.

To estimate the vector of unknowns $\boldsymbol{\beta}$ we formulate the following regularized robust least squares optimization problem

$$\min_{\boldsymbol{\beta}} H\Big[\mathbf{y} - Z\boldsymbol{\beta}\Big] + \lambda||\boldsymbol{\beta}||_2^2, \qquad \text{where} \quad H_i\left[r_i\right] = \begin{cases} r_i^2 & \text{if } |r_i| \leq K \\ K\left(2|r_i| - K\right) & \text{if } |r_i| > K \end{cases}. \tag{3.4}$$

$H_i\left[r_i\right]$ is the Huber loss function [1], $r_i = \left[\mathbf{y} - Z\boldsymbol{\beta}\right]_i$ for $i = 1, \ldots, n(T-1)$ is the residual, K is a tuning threshold influencing resistance to outliers, usually selected as a multiple of the standard deviation of residuals, and $\lambda > 0 \in \mathbb{R}$ is the regularization parameter. Figure 3.2 shows an example of the Huber loss function for $K = 5$. As can be seen, whenever the absolute value of the residuals are smaller than K, the applied penalty is quadratic, however, for the residuals exceeding K or $-K$ only linear penalty is applied. In this way, the outliers with large residual values do not have too much influence on the resulting solution.

The reason we have used the robust form of the least squares is to prevent possible data noise and outliers from distorting the computed solution, which can happen if a

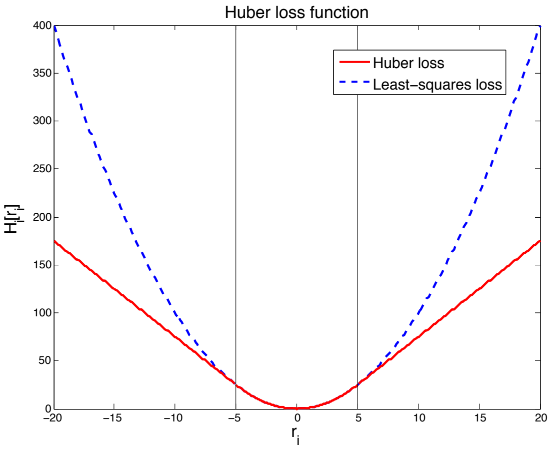


Figure 3.2: Illustration of the Huber loss function [1]. Blue dashed line represents quadratic cost r_i^2 and red line is the Huber loss. Vertical black lines show the constants $-K$ and K ($K = 5$), which mark the transition of the loss function from quadratic to linear penalty.

simple quadratic cost function is used instead. It can be shown [84] that (3.4) can easily be solved using regularized iterative re-weighted least squares

$$\min_{\boldsymbol{\beta}} ||W\left(\mathbf{y} - Z\boldsymbol{\beta}\right)||_2^2 + \lambda||\boldsymbol{\beta}||_2^2, \tag{3.5}$$

where W is the diagonal weighting matrix, whose i-th diagonal element is

$$W_{ii}(r_i) = \begin{cases} 1 & \text{if } |r_i| \leq K \\ \frac{K}{|r_i|} & \text{if } |r_i| > K \end{cases}.$$

Observe also that we have included a regularization parameter λ to improve generalization of the constructed model [85] as well as to prevent possible ill-conditioning of the matrix Z, which can lead to inaccurate solution $\boldsymbol{\beta}$. The ill-conditioning usually happens in cases when time series, representing sensor measurements, become highly correlated among each other, leading to rank-deficient Z. Finally, note that matrix Z in (3.5) can become very tall in cases when T is large and standard approaches of estimating $\boldsymbol{\beta}$ based on regular QR decomposition [86] become impractical. For this purpose, in practice, we use the approach of [87] based on Tall and Skinny QR (TSQR), which enables to perform QR of a tall matrix in a block-by-block sequential manner.

Note that instead of the least squares regression based on Huber loss function, one can employ other robust methods, for example, least median squares [88] or least trimmed squares regression [2]. The main advantage of these type of algorithms is that they achieve the highest breakdown point of 50%, i.e., the minimum percentage of data which needs to be corrupted to make the regression technique to break down. On the other hand, these approaches are known to be computationally expensive, whose complexity grows exponentially with the problem dimensionality. In practice, their solutions are usually obtained using various heuristics, which are based on the search of the subsets of data that minimize the optimization criterion [88]. In Section 3.4 we present comparison study, which showed that for the considered anomaly detection problem, the least squares regression based on Huber loss still performs better than the other alternatives.

3.2.2 Distance matrix

Having computed the models for each flight, the next step in the proposed anomaly detection framework is to construct the distance matrix $D \in \mathbb{R}^{N \times N}$ among the N flights. The idea is based on computing each element D_{ij} using the residuals of the model built on the data of flight i evaluated on the data from flight j, for $i = 1, \ldots, N$ and $j = 1, \ldots, N$.

Specifically, let $\widehat{A}^i$ and $\widehat{B}^i$ be the estimated parameters of the model in (3.2), computed using data from flight i. Evaluate this model on data from flight j by computing the residuals $r_k^{ij} = y_k^j - \widehat{y}_k^i$, where $\widehat{y}_k^i = \widehat{A}^i y_{k-1}^j + \widehat{B}^i u_{k-1}^j,\ k = 2, \ldots, T_j$, is the one-step prediction of the estimated model and y_k^j, u_k^j are the data of flight j of length T_j. Note that in general $r_k^{ij} \neq r_k^{ji}$. Next, we utilize the computed residuals r_k^{ij} to construct a scalar dissimilarity measure D_{ij} between flight i and j. The idea is to treat the residuals $r_k^{ij} \in \mathbb{R}^n$ as $T_j - 1$ vectors in n-dimensional space and compute their center of mass. Intuitively, the closer the point to this center, the more likely it represents the distance between similar flights. Measuring the closeness to the mean using Euclidean distance has a drawback in that it assumes that the points are distributed in spherical manner around the center, which is usually not the case in many practical scenarios. A better approach is to use Mahalanobis distance [89], which is a generalization of the Euclidean distance, and it accounts for the variance along each dimension as well as the covariance between the dimensions, thus, more accurately measuring the proximity to the mean.

Thus, at each time stamp we compute

$$m_k^{ij} = \sqrt{(r_k^{ij} - \mu_{r_k^{ij}})^T C^{ij^{-1}} (r_k^{ij} - \mu_{r_k^{ij}})}, \tag{3.6}$$

where C^{ij} is the sample covariance matrix

$$C^{ij} = \frac{1}{T_j - 1} \sum_{k=2}^{T} (r_k^{ij} - \mu_{r_k^{ij}})(r_k^{ij} - \mu_{r_k^{ij}})^T,$$

and $\mu_{r_k^{ij}} = \frac{1}{T_j - 1} \sum_{k=2}^{T} r_k^{ij}$ is the sample mean. Finally, the dissimilarity measure D_{ij} is then computed by combining the m_k^{ij}'s, e.g., using the standard deviation $D_{ij} = \frac{1}{T_j - 1} \sum_{k=2}^{T} (m_k^{ij} - \mu_{m_k^{ij}})^2$, where $\mu_{m_k^{ij}}$ is the mean.

Besides the standard deviation, other summary statistics can also be used but we observed that it performed well in practice, adequately capturing variability in the residuals. Specifically, during experiments we noticed that for the flights i and j which are similar, the residuals usually stayed small throughout the flight. On the other hand, when comparing normal and anomalous flights, the residuals also remained small except for some segments which contained large deviations (e.g., see right upper plot in Figure 3.2). Detection of such flights can be viewed as a separate outlier detection problem in one dimensional time-series, in which one can use various sophisticated approaches, e.g., based on support vector regression [90], using mixture transition distribution approach of [91] or using median information from the neighborhood [92] to identify outliers. However, we found that a summary statistic such as standard deviation, which uses a sum of quadratic deviations, is sufficiently sensitive to outliers and had a good performance in our experiments, therefore can serve as an adequate dissimilarity measure D_{ij}. Note also that the distance matrix D obtained in this way is not symmetric, however, it can be symmetrized in a number of ways [78], e.g., by averaging $D_{ij} = \frac{1}{2}(D_{ij} + D_{ji})$.

3.2.3 Anomaly detection

The estimated distance matrix can now be used to detect outliers, which correspond to the anomalous flights in our case. For this purpose we utilize the local outlier factor (LOF) approach of [3]. Intuitively, LOF is based on comparing the local density of a point with the density of its neighbors using the pairwise distances between the points.

Specifically, LOF proceeds by computing for a given constant k and distance matrix D, the set of k-nearest neighbors for each flights F^i, $i = 1, \ldots, N$. Denote this set as $S_{NN_k}(F^i)$. The distance from flight F^i to its k-th nearest neighbor is denoted as $d_k(F^i)$. Next, we define a reachability distance *from* flight F^j to flight F^i to be

$$rd_k(F^i, F^j) = \max(d_k(F^j), D_{ij}),$$

i.e., it is an actual distance between two flights but at least $d_k(F^j)$, so that all flights within a set $S_{NN_k}(F^j)$ are treated as equidistant. Using this information we can now compute local reachability density of flight F^i

$$lrd(F^i) = 1/\left(\frac{\sum_{f \in S_{NN_k}(F^i)} rd_k(F^i, f)}{|S_{NN_k}(F^i)|}\right).$$

3.3 Overview of Compared Algorithm

In this Section we present an overview of the algorithms, which we used in the comparison study in Section 3.4 to evaluate the proposed algorithm in detecting aviation safety events. The four baseline algorithms we considered were the MKAD, the current state of the art approach in detecting anomalous flight events, which was used in two versions, one based on continuous data only and one with a mixture of discrete and continuous parameters. And the two approaches based on DTW, one of them is based on voting and the other based on the covariance weighting.

3.3.1 Dynamic Time Warping

Dynamic time warping (DTW) [93] is a popular method to optimally align two univariate time series of possibly unequal length by warping each of them until they match. The size of the smallest alignment (or warping path) is then considered to be the distance between two sequences. Note that the computation of the alignment is based on some local distance measure, which compares an element from one sequence to an element from another sequence, and here we assume that such measure is a Euclidean distance. For our problem, we let $F^i_k = \left\{y^i_k, u^i_k\right\}$, $k = 1, \ldots, T_i$ and $F^j_l = \left\{y^j_l, u^j_l\right\}$, $l = 1, \ldots, T_j$ denote the multivariate time series data from flights i and j. Denote by f, $f = 1, \ldots, n + m$, the index of a specific dimension in the time series. Then the

DTW distance between two sequences is denoted as $DTW\Big[F^i(f), F^j(f)\Big]$ and the local distance measure is of the form $\Big(F_k^i(f) - F_l^j(f)\Big)^2$ for all k and l.

Since DTW was proposed for univariate sequences but our work is concerned with comparison of multivariate time series, in what follows, we propose two extensions, enabling anomaly detection among multivariate time series based on DTW. In these approaches, we follow the main idea of the proposed framework in Fig. 3.1 and construct distance matrix using DTW rather than VARX. Applying the LOF method, discussed in Section 3.2.3, to such a matrix reveals the anomalous flights.

Anomaly Detection using Vote-based DTW

In the vote-based DTW, we construct $m + n$ distance matrices D^f between the flights corresponding to each feature $f = 1, \ldots, n + m$, i.e., $D^f_{i,j} = \frac{1}{T_i+T_j} DTW\Big[F^i(f), F^j(f)\Big]$. We then apply LOF to each distance matrix D^f, resulting in $n + m$ lists of anomaly scores for each of N flights, sorted in decreasing order, so that the top flight is the most anomalous. The final score is then decided based on the voting, i.e., flight i is considered anomalous if it was flagged τ times as anomalous. The number $\tau \in (1, \ldots n + m)$ is determined empirically and in our experiments we used $\tau = \frac{n+m}{2}$, i.e., a majority-based voting.

We note that an alternative approach is to combine $n + m$ matrices D^f (e.g., by averaging) and then apply LOF on the resulting matrix to identify outliers. However, this approach might decrease the chances of identifying anomalies because the combination of $n+m$ matrices D^f can wash out the extreme values D^f_{ij}, thus hiding potentially anomalous flights.

Anomaly Detection using Covariance-based DTW

Using the ideas from [94] we propose the covariance-based DTW. Recall that in a univariate DTW we use a scalar local distance measure based on Euclidean distance $\Big(F_k^i(f) - F_l^j(f)\Big)^2$. In the covariance-based DTW, we propose to use a weighted vector-based distance measure, i.e., $\|F_k^i - F_l^j\|_W^2$, where W is a weighting matrix. A possible choice for W can be a matrix constructed based on the inverse covariance of the time series F^i and F^j. Specifically, let C_{F^i} denote a covariance of multivariate time series

F^i and C_{F^j} be a covariance of F^j, then we set $W = (C_{F^i} + C_{F^j})^{-1}$.

The anomaly detection procedure then proceeds as follows. For each pair of flights, we compute a distance matrix using multivariate DTW, i.e., $D_{i,j} = \frac{1}{T_i+T_j} DTW_W \left[F^i, F^j\right]$ based on local distance measure $\|F^i_k - F^j_l\|^2_W$ and then, as before, apply LOF to identify the anomalous flights.

3.3.2 Multiple Kernel Anomaly Detection

The Multiple Kernel Anomaly Detection (MKAD) [74] is designed to detect anomalies in the heterogeneous multivariate time series, where both discrete and continuous features are present. The ability to incorporate discrete features is advantageous for anomaly detection since it enables modeling switching sequences of the flight and the order of the switching can provide additional information to identify abnormal system behavior. Specifically, if we assume that the time series F^i and F^j now include both continuous and discrete sequences, then the operation of MKAD can be described as follows. First, construct the kernel of the form $K\Big(F^i(f), F^j(f)\Big) = \alpha K_d\Big(F^i(f), F^j(f)\Big) + (1-\alpha)K_c\Big(F^i(f), F^j(f)\Big)$, where K_d is a kernel over discrete sequences and K_c is a kernel over continuous time series and $\alpha \in [0, 1]$ is a weight, which is usually set to $\alpha = 0.5$. For discrete sequences, the normalized longest common subsequence (LCS) is used, i.e., $K_d\Big(F^i(f), F^j(f)\Big) = \frac{|LCS(F^i(f),F^j(f))|}{\sqrt{T_iT_j}}$, where $|LCS(F^i(f), F^j(f))|$ denotes the length of LCS. For continuous sequences, the kernel $K_c\Big(F^i(f), F^j(f)\Big)$ is inversely proportional to the distance between symbolic aggregate approximation (SAX) representation [75] of continuous sequences $F^i(f)$ and $F^j(f)$. The constructed kernel $K \in \mathbb{R}^{N\times N}$, where N is the number of flights, is then used in one-class support vector machine (SVM) [76] to construct a hyperplane to separate rarely seen (anomalous) flights from frequently seen (normal) flights. One-class SVM adapts the traditional SVM methodology to the one-class classification problem. In particular, after transforming the flight time series via kernel to a high-dimensional feature space, the algorithm treats the origin as the only member of the anomalous class. A hyperplane is then constructed to maximally separate the data from the origin. Consequently, the flights which are located on that side of the boundary closest to the origin are classified as anomalous while all other flights are treated as normal.

In our testing procedures, we employed two versions of MKAD, one with $\alpha = 0$, which corresponds to using continuous features only and one with $\alpha = 0.5$, corresponding to an algorithm capable of dealing with heterogeneous data.

3.4 Experiments

In this section we present the evaluation results of the proposed framework on the FOQA flight dataset from a partner airline company, containing over a million flights, each having a record of about 300 parameters, including sensor readings, control inputs and weather information. We have selected flights with landings at the same destination airport and the aircrafts of the same fleet and type, so that we eliminate potential differences related to aircraft dynamics or landing patterns. Data analysis focused on a portion of the flight below 10000 feet until touchdown, corresponding to the approach and landing phases, usually having the highest rates of accidents [95].

We evaluated the proposed algorithm using two methodologies. In the first one (Section 3.4.1), using information provided by the airline company's exceedance-based algorithm, we picked a set of flights with known anomalous events and a set of flights containing no such events. Knowing data labels, we evaluated the performance of the algorithm quantitatively, using receiver operating characteristic (ROC) analysis [96]. In the second approach (Section 3.4.2), we tested the framework in a more realistic scenario when there is no information about which flights are normal or anomalous. The presented analysis is only qualitative since no ground truth is available and the discoveries were validated by the domain experts, including a retired pilot with over 35 years of flying experience.

The performance of our VARX-based anomaly detector was compared with MKAD [74] for continuous data and for heterogeneous data, as well as with two methods based on DTW, i.e., the vote-based DTW and the covariance-based DTW. Out of 300 parameters originally present in the dataset, we have selected 54 continuous features for VARX, DTW and MKAD, while for MKAD for heterogeneous data we additionally included 23 discrete parameters. We have implemented the proposed algorithm and DTW in Python and the framework's easily parallelizable structure was exploited by distributing the computations across the computer cluster with up to 1800 cores. In all the experiments we have used a first-order VARX model for flight representation. The implementation of MKAD was provided by the NASA colleagues.

We note that the standard assumption in VARX modeling is the stationarity of the data. Specifically, this requires a constant mean and variance of each time series while covariance should depend only on the time difference between two time stamps and independent of the shift along the time series. In practice, however, this is rarely satisfied and the data exhibits non-stationarity. A popular method to introduce stationarity is to perform differencing of individual time series [97]. Usually, first or second order differencing suffices and a practical criteria to check the stationarity is to compute the autocorrelation function and ensure that it damps down quickly. In all of our experiments, before building the VARX model, we normalized individual time series by subtracting mean and applying first order differencing.

3.4.1 Labeled data

In this study, we have selected 10 flights which had high pitch rate at landing (denote these flights as D_1), 10 flights with a go-around event (denote as D_2), 10 flights with a large vertical acceleration at touch down (denote as D_3) and 100 anomaly-free flights (D_4). The D_1 flights have fast angle change of the aircraft's nose and are considered operationally significant since this can lead to a bouncing on the runway or tail strike, causing significant structural damage and threaten flight safety. The D_2 flights are the ones which abort their normal landing, fly back up to a certain altitude and try to repeat the landing again. These flights are considered operationally significant anomalies since they could be executed in response to an emergency or unsafe conditions in the air or on the runway. Finally, the D_3 flights are also of interest since large vertical acceleration at the moment of contact with runway could be due to hard landing, which are also operationally significant events.

Anomaly: High pitch rate at landing

The results of detecting high pitch rate flights in the dataset consisting of D_1 and D_4 flights are shown in Figure 3.1. It can be seen that the VARX algorithm performed better as compared to others with DTW algorithm based on voting having the worst performance. The two right plots in Figure 3.1 examine one of the anomalous flights as it was detected by VARX algorithm. The residuals remained small but after the time stamp 450 they started increasing, signaling the abnormal behavior. In the corresponding time series at the bottom we can notice a high increase and drop of the pitch angle (greater than $3°/sec$) right before the touch down.

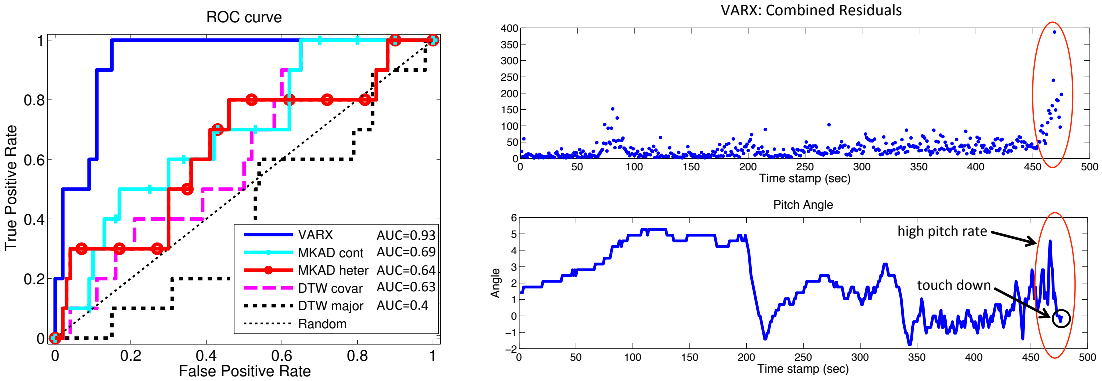


Figure 3.1: Detection of high pitch rates at landings. The left plot shows the ROC curve for detecting 10 anomalous in 110 flights, with 100 of them being normal. The AUC for VARX is 0.93, AUC for MKAD continuous is 0.69, AUC for MKAD heterogeneous is 0.64, AUC for DTW covariance is 0.63 and finally AUC for DTW majority is 0.4. The right two figures show the example of VARX-based algorithm detecting abnormal behavior. The top plot shows the combined residuals during one of the anomalous flights and the bottom figure shows the corresponding history of pitch angle measurements with markings of the anomalous segment.

Anomaly: Go-around

Figure 3.2 shows the accuracy of detecting the go-around flights in the dataset containing D_2 and D_4 flights. Observe that for this type of anomalies the AUC for all five methods is higher as compared to the flights with high pitch rate at landings. This occurred since several flight parameters deviated significantly from their normal behavior during the go-around event, with the deviation being more pronounced, thus easing the detection task. An example of such flight is shown at the right side of Figure 3.2, where around time stamp 600 the altitude increases to about 4000 feet. The corresponding model's residuals for VARX method are also shown which have a sharp jump when the go-around is initiated.

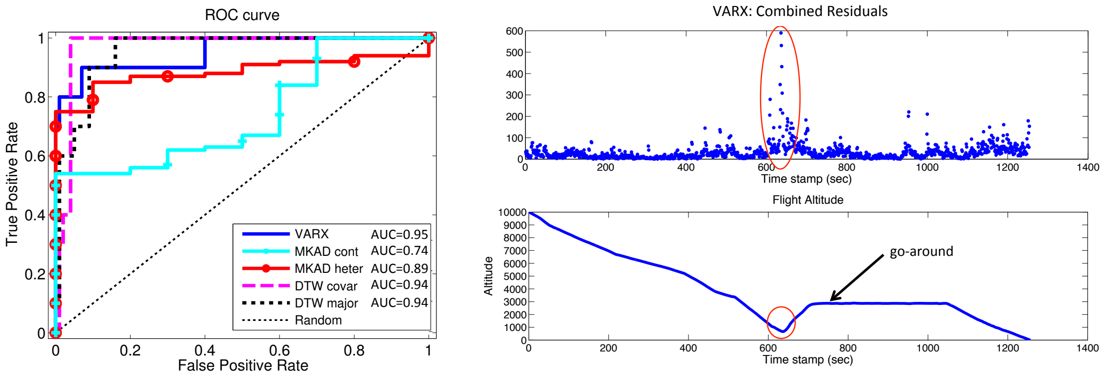


Figure 3.2: Detection of the go-around flights. Left plot shows the ROC curve anomaly scores for detecting 10 anomalous in 110 flights, with 100 of them being normal. The AUC for VARX is 0.95, AUC for MKAD continuous is 0.74, AUC for MKAD heterogeneous is 0.89, AUC for DTW covariance is 0.94 and finally AUC for DTW majority is 0.94. The right plots show the combined residuals for one of the anomalous flights as detected by VARX and the corresponding flight altitude with markings showing the start of the go-around event.

Anomaly: Large vertical acceleration at touch down

Finally, the accuracy of detecting the flights with large vertical acceleration at touch down in the dataset containing D_3 and D_4 flights are shown in Figure 3.3. It can be seen that the performance of VARX-based method was slightly better than DTW and MKAD with DTW covariance-based approach performing the worst. The right plots show the detection of one of the anomalous flights by VARX algorithm. Throughout the landing, the combined residuals remained low. However, during the touch down, the vertical acceleration increased rapidly, possibly indicating a hard landing, which led to a spike in the residuals and thus this flight was flagged as anomalous.

Comparison studies

In this Section we also present the results of the comparison studies which justify several design choices we have made earlier in Section 3.2.

VARX estimation losses. Using the labeled data representing the same three types of anomalies as before, we compared three approaches for estimating the VARX parameters: least squares based on robust Huber loss [1], ordinary least squares (OLS)

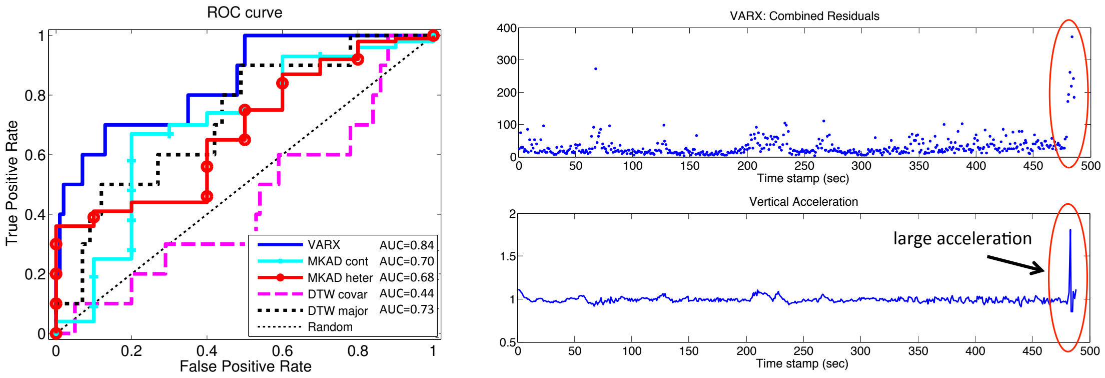


Figure 3.3: Detection of large vertical acceleration at touch down. ROC curve for detecting 10 anomalous in 110 flights, with 100 of them being normal is shown in left plot. The AUC for VARX is 0.84, AUC for MKAD continuous is 0.70, AUC for MKAD heterogeneous is 0.68, AUC for DTW covariance is 0.44 and finally AUC for DTW majority is 0.73. The right top plot shows the combined residuals for VARX algorithm during one of the anomalous flights and the bottom figure shows the corresponding history of acceleration measurements with red oval showing the anomaly.

and least trimmed squares regression (LTS) [2]. Table 3.1 shows the AUC scores for the three datasets for each of the estimation methods. As can be seen, OLS

	High pitch rate at landing	Go-around	Large vert. accel. at touch down
OLS	0.950	0.980	0.851
Huber	0.952	0.957	0.863
LTS	0.914	0.921	0.783

Table 3.1: Comparison of three algorithms (OLS, Huber-based [1] regression and least trimmed squares regression (LTS) [2]) for VARX parameter estimation on each of the three labeled flight datasets, each consisting of 110 flights (100 normal and 10 anomalous). The results are shown in terms of the anomaly detection performance using the area under ROC curve (AUC) scores. LOF method was used as the anomaly detector.

and Huber-based least squares performed similarly with the method using Huber loss achieving slightly more accurate results on two of the datasets. On the other hand, the estimation based on LTS was less accurate, which could be explained by the fact that the considered VARX estimation problem is high dimensional and the algorithm involves considerable combinatorial search [2].

Density-based anomaly detection methods. We have also performed experiments to justify our choice of LOF as the outlier identification technique in our anomaly detection framework. In particular, after computing the flight-by-flight distance matrix, in the final step of our framework (see Figure 3.1) we applied three alternative anomaly detectors and compared the resulting detection accuracy. The three considered methods were LOF, DBSCAN [4], a popular density-based clustering algorithm, and an approach based on nearest-neighbor (NN) [5], in which flight's abnormality is determined by the distance to the first nearest neighbor in the flight-by-flight neighborhood graph. We tested these methods on the same three labeled flight datasets as before, and the results are shown in Table 3.2, where we have used prediction accuracy as the measure of performance. It can be seen that all the methods performed similar to each other with the LOF achieving slightly better results than the others on two of the datasets.

	High pitch rate at landing	Go-around	Large vert. accel. at touch down
DBSCAN	0.918	0.973	0.927
NN	0.927	0.982	0.945
LOF	0.936	0.983	0.936

Table 3.2: Comparison of three anomaly detection algorithms (LOF [3], DBSCAN [4] and nearest-neighbor (NN) based approach [5]) on each of the three labeled flight datasets, each consisting of 110 flights (100 normal and 10 anomalous). The results are shown in terms of the anomaly detection performance using $accuracy = \frac{true\ positive\ +\ true\ negative}{positive\ +\ negative}$. Huber-based loss function was used to estimate VARX parameters.

3.4.2 Unlabeled data

In this study we selected 20000 flights with no information available about which of them are normal and anomalous. We tested the proposed VARX-based algorithm and compared its performance with the other four methods. For each method, we examined the top 100 flights with the highest anomaly scores to determine the flights containing operationally significant events. In Table 5.2 we present a summary of the discovered anomalies, which were also examined and validated by the experts.

VARX	MKAD continuous	MKAD heterogeneous	DTW covariance
go-around (35)	go-around (10)	go-around (17)	go-around (30)
high speed in approach (6)	high speed in approach (2)	high pitch at touch down (1)	delayed braking (1)
low pitch at landing (1)	delayed braking at landing (1)	high speed in approach (2)	high rate of descent (1)
bounced landing (1)	high rate of descent (3)	low speed at touch down (1)	bank cycling in approach (1)
delayed braking at landing(1)	bank cycling in approach(1)	low path in approach (1)	bounced landing (1)
high path in descent (1)	high pitch at touch down (1)	flaps retracted in approach (1)	DTW majority
high pitch at touch down (19)	autoland warning (3)	unusual usage of AP (26)	go-around (37)
holding pattern (3)	short flare time (4)	unusual usage of FD (27)	high speed in approach (2)
altitude deviation (1)			high pitch at landing (1)
wake turbulence (1)			bank cycling (1)

Table 3.3: The anomalies discovered in the top 100 anomalous flights, ranked by each of the five anomaly detection methods in the set of 20000 unlabeled flights.

Discussion of the results

Among the top 100 flights, we found that the most common type of anomaly was the go-around flights. These results confirm our earlier tests where we established high detection accuracy of this type of flights. In total there were 61 go-arounds in the examined set of flights and although MKAD could only detect 10 of them using continuous features and 17 based on both types of data, the other approaches identified over 30 such flights, with VARX and DTW vote-based methods detecting the largest number of them. Figure 3.4 shows the scores for the four approaches, where the red circles mark the go-around flights. It can be seen that although all methods placed these flights in the upper part of their anomaly lists, the VARX-based and DTW approaches detected them with higher accuracy as compared to MKAD.

On the other hand, after examining other non-go-around flights from the MKAD output, we found a number of operationally significant anomalies, which are discussed next. The detected anomalous flights which had high speed in approach or high pitch at touch down and some of the go-around flights were the same for continuous and heterogeneous MKAD. On the other hand, due to the use of discrete parameters (various autopilot and guidance system modes, not used in the other methods) heterogeneous MKAD also detected 26 flights which used flight path angle, a rarely used vertical autopilot mode, and 27 flights where the flight director was turned off for over 2 minutes during the approach, which is an unusual behavior since, typically, flight director is used throughout the approach to assist the pilot with vertical and horizontal cues even when the autopilot is not engaged. Moreover, presence of discrete flight parameters improved MKAD performance in detecting additional go-around flights as compared to a scenario

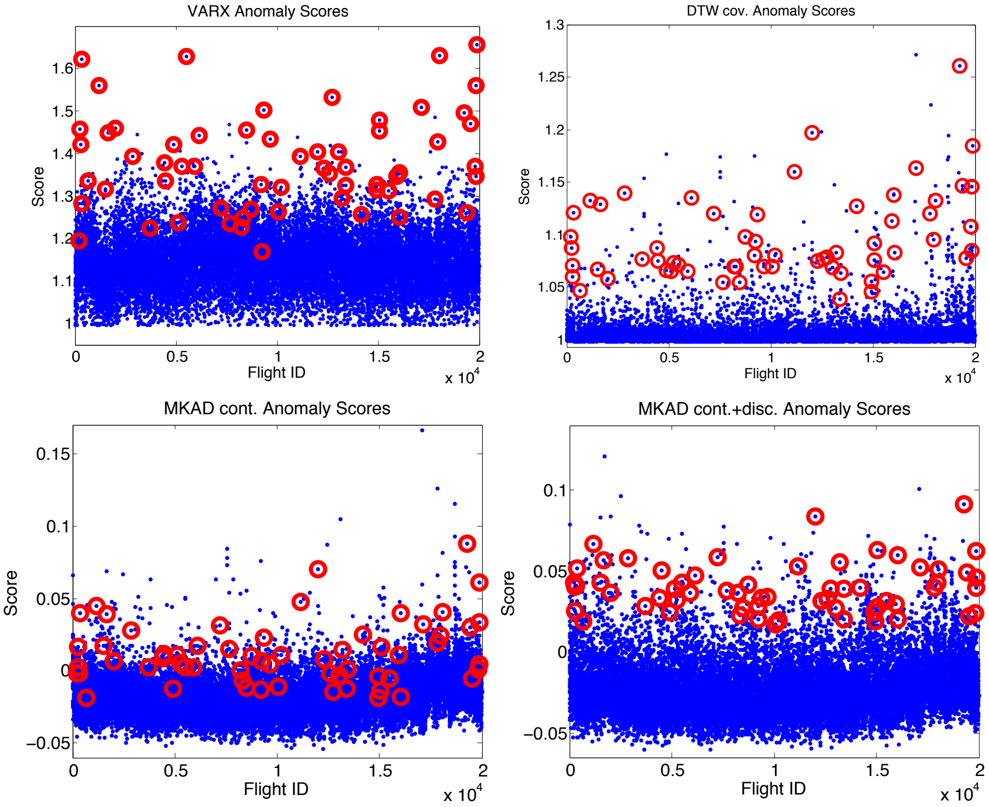


Figure 3.4: Distribution of the anomaly scores for 20000 flights computed by VARX algorithm (upper left), DTW covariance-based approach (upper right) and MKAD based on continuous parameters (lower left) and heterogeneous data (lower right). Red circles in all plots denote all 61 go-around flights in the selected 20000 flights.

when only continuous features are used.

The anomaly detection based on DTW had a good performance in detecting go-around flights. All the discovered events had a common feature of being anomalous in a single parameter, thus missing more complex events which were better detectable by VARX and MKAD approaches.

The anomalous flights detected by VARX-based algorithm had abnormal events containing in a single parameter, such as go-arounds, high speed in approach, high pitch rate, etc., as well as in multiple features, such as altitude deviation and wake turbulence. In the following Section we discuss in details two examples of the previously unknown anomalies involving multiple parameters.

Previously unknown anomalies detected by VARX method

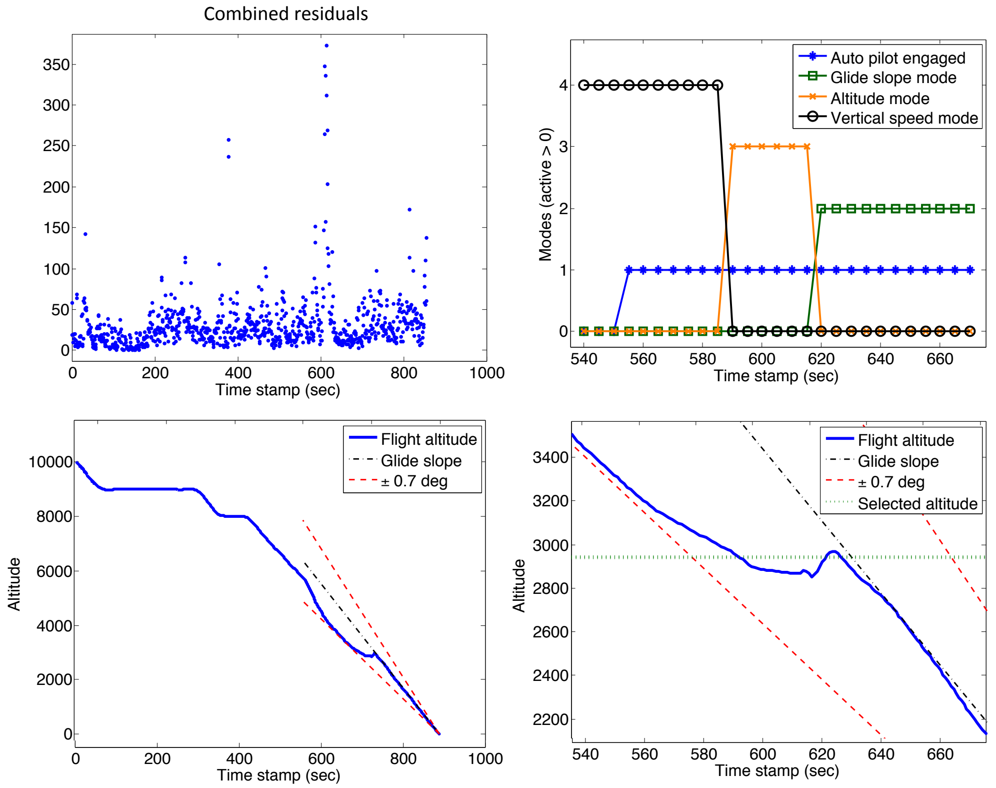


Figure 3.5: Altitude deviation anomaly. The left column shows the distribution of residuals during the flight and the altitude profile, with glide slope for reference. The right column presents the zoomed-in part of the flight. The top plot shows few key control modes during the event and the bottom one shows the corresponding flight altitude. Note that the control modes are shown for reference and were not used in VARX-based approach.

Altitude Deviation. Figure 3.5 shows the flight that had altitude deviation anomaly, which we explain next. From the upper left plot, showing the history of the residuals, we can see that the event occurred around time stamp 600, the time when the aircraft was capturing the glide slope (see lower left plot). Now examining the two plots on the right, we can see that the plane was descending to the selected altitude of 3000 feet, however it was not leveling as expected and the pilot engaged the altitude hold mode too late with the aircraft being well below the required altitude. Around time stamp 615 the aircraft started the ascent to correct the altitude discrepancy. At this time altitude hold mode was switched off and glide slope mode was turned on. However, since the inertia was too high the airplane continued climbing for a few seconds and then immediately started descending as it captured the glide slope. This part of the flight is associated with abrupt acceleration and deceleration, which usually leads to an uncomfortable experience for the passengers.

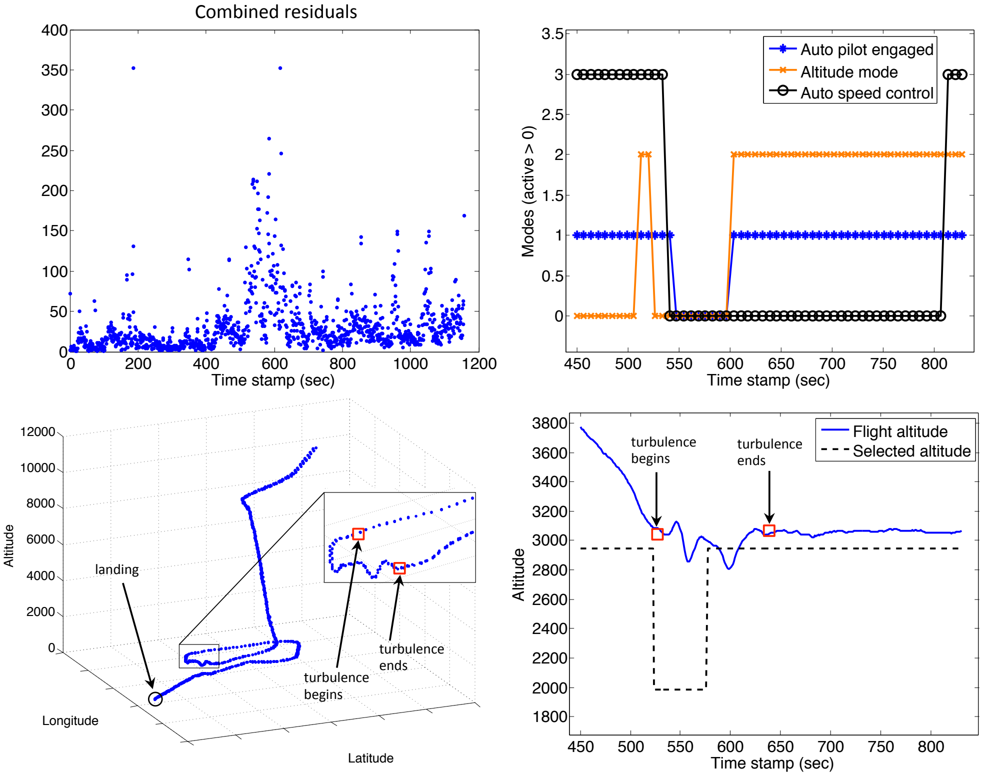


Figure 3.6: Wake turbulence anomaly. Left column shows combined residuals across the flight and the corresponding trajectory with markings of landing and anomalous segment. Right column shows few key control modes during the event and the corresponding zoomed-in part of the flight altitude. Note that the control modes are shown for reference and were not used in VARX-based approach.

Wake turbulence. Another example of the discovered operationally significant event is shown in Figure 3.6 where the flight experienced a wake turbulence anomaly. It can be seen that the event happened in the $550 - 650$ seconds time range and is marked by the square on the flight trajectory. The aircraft was in the final approach phase with the selected altitude reduced from 3000 to 2000 feet. At this time the pilot engaged the altitude hold mode but after about 20 seconds the aircraft experienced large swings in altitude forcing the pilots to turn off altitude hold mode, followed by disengagement of autopilot and auto speed controls. Once the aircraft was diverted to a holding pattern, the turbulence stopped. On the second pass there was no sign of turbulence and the airplane landed. Our hypothesis of a turbulence is also reinforced by the fact that the flight occurred in the evening, around 8 pm, which is usually a time of increased traffic volume. The turbulence may have been due to a preceding aircraft. However, FOQA data does not contain any information about the surrounding aircrafts.

Discussion of the results

The above analysis showed that the proposed VARX-based approach can be considered as complementary to the MKAD algorithm. Our method is particularly suited for the detection of anomalies which are accompanied with rapid changes in the parameters, e.g., go-around flights marked by fast acceleration and engine spool-up, or flights which have high pitch rates at landing, etc. On the other hand, the proposed method is prone to miss anomalies manifested in abnormal behavior of the discrete features, e.g., unusual sequence of autopilot modes, which are better detected by MKAD algorithm utilizing its kernel over discrete sequences using LCS. On the other hand, the comparison with the other baseline algorithms based on DTW revealed the advantage of VARX-based approach, which discovered previously unknown, complex anomalous events involving multiple parameters. The VARX modeling naturally exploits the correlation between the features, which is not achieved by a simple techniques based on DTW, whose discovered anomalies usually were caused by a single parameter.

Chapter 4

Estimating Structured Vector Autoregressive Model

In this chapter we study the properties of the VAR estimation problem. Recall that in the previous chapter one of the key steps in the proposed anomaly detection approach, shown in Figure 3.1, was the construction of the VAR model for each flight, where the corresponding estimation problem was shown in (3.5). In that formulation the matrix Z and vector y are composed from the output of the VAR model (3.2), which causes the correlations between the rows and columns of matrix Z and rows of vector $\mathbf{y}$. Existing theoretical results, e.g., [98], [99], which establish sample complexity and bounds on the estimation error of a solution of linear regression problems like (3.5) do not hold, since they rely on independent and identically distributed data samples. In this chapter we present results for characterizing sample complexity and error bounds in estimating structured vector autoregressive models. In particular, in Section 4.2 we present the estimation problem for the structured VAR model. The main results on the estimation guarantees are established in Section 4.3. The experimental results testing the derived bounds are shown in Section 4.4.

4.1 Introduction

To estimate parameters of VAR (or VARX) model, one usually transforms it into an appropriate set of linear equations and then solves a linear regression formulation using regularized least-squares problem, e.g., ridge regression [100] or lasso [30]. The statistical guarantees such as sample complexity and error bounds of the estimated solution were developed under the conditions of independent and identically distributed samples [98], [99]. However, since in our case the data are coming from vector autoregressive process, the samples are spatially and temporally correlated. Therefore, there is a need to establish guarantees in the case of computing VAR models using regularized estimators. In our work we show that the sample complexities and error bounds are, somewhat surprisingly, of the same order as if the samples were independent. The constants in the order are of course different and rely on the covariance structures of the noise as well as the characteristics of the time series model.

In recent literature, the problem of estimating structured VAR models has been considered for the special case of L_1 norm. For example, [101] analyzed a constrained estimator based on the Dantzig selector [102], and established the recovery results for the special case of L_1 norm. [103] considered a regularized VAR estimation problem under Lasso and Group Lasso penalties and derived oracle inequalities for the prediction error and estimation accuracy. However, their analysis is for the case when the dimensionality of the problem is fixed with respect to the sample size. Moreover, they employed an assumption on the dependency structure in the VAR, thus limiting the sample correlation issues mentioned earlier. The work of [104] studied regularized Lasso-based estimator while allowing for problem dimensionality to grow with sample size, utilizing suitable martingale concentration inequalities to analyze dependency structure. [105] considered L_1 VAR estimation for first order models ($d = 1$) assuming $\|A_1\|_2 < 1$, and the analysis was not extended to the general case of $d > 1$. In recent work, [106] considered a VAR Lasso estimator and established the sample complexity and error bounds by building on the prior work of [105]. Their approach exploits the spectral properties of a general VAR model of order d, providing insights on the dependency structure of the VAR process. However, in line with the existing literature, the analysis was tailored to the special case of L_1 norm, thus limiting its generality.

Compared to the existing literature, our results are substantially more general since the results and analysis apply to *any* norm $R(\cdot)$. One may wonder—given the popularity of L_1 norm, why worry about other norms? Over the past decade, considerable effort has been devoted to generalize L_1 norm based results to other norms [107, 108, 109, 110].

Our work obviates the need for a similar exercise for VAR models. Further, some of these norms have found key niche in specific application areas e.g., [111, 112]. From a technical perspective, one may also wonder—once we have the result for L_1 norm, why should not the extension to other norms be straightforward? A key technical aspect of the estimation error analysis boils down to getting sharp concentration bounds for $R^*(Z^T\epsilon)$, where $R^*(\cdot)$ is the dual norm of $R(\cdot)$, Z is the design matrix, and ϵ is the noise [109]. For the special case of L_1, the dual norm is L_∞, and one can use *union bound* to get the required concentration. In fact, this is exactly how the analysis in [106] was done. For general norms, the union bound is inapplicable.

Our analysis is based on a considerably more power tool, *generic chaining* [113], yielding an analysis applicable to any norm, and producing results in terms of geometric properties, such as Gaussian widths [114], of sets related to the norm. Results for specific norms can then be obtained by plugging in suitable bounds on the Gaussian widths [115, 116]. We illustrate the idea by recovering known bounds for Lasso and Group Lasso, and obtaining new results for Spare Group Lasso and OWL norms. Finally, in terms of the core technical analysis, the application of generic chaining to the VAR estimation setting is not straightforward. In the VAR setting, generic chaining has to consider a stochastic process derived from sub-exponential martingale difference sequence (MDS). We first generalize the classical Azuma-Hoeffding inequality applicable to sub-Gaussian MDSs to get an Azuma-Bernstein inequality for sub-exponential MDSs. Further, we use suitable representations of Talagrand's γ-functions [113] in the context of generic chaining to obtain bounds on $R^*(Z^T\epsilon)$ in terms of the Gaussian width $w(\Omega_R)$ of the unit norm ball $\Omega_R = \{u \in \mathbb{R}^{dp} | R(u) \leq 1\}$. Our estimation error bounds in the VAR setting are *exactly of the same order* as Lasso-type models in the i.i.d. setting implying, surprisingly, that the strong temporal dependency in the VAR setting has no adverse effect on the estimation.

Without the loss of generality, we consider a VAR model of order d with no exogenous variables, i.e., a model of the form, where we changed slightly our notations as compared to the ones used in Section 3.2.1

$$x_t = A_1 x_{t-1} + A_2 x_{t-2} + \cdots + A_d x_{t-d} + \epsilon_t \ , \tag{4.1}$$

where $x_t \in \mathbb{R}^p$ denotes a multivariate time series, $A_k \in \mathbb{R}^{p \times p}, k = 1, \ldots, d$ are the parameters of the model, and $d \geq 1$ is the order of the model. In this work, we assume that the noise $\epsilon_t \in \mathbb{R}^p$ follows a Gaussian distribution, $\epsilon_t \sim \mathcal{N}(0, \Sigma)$, with $\mathbb{E}(\epsilon_t \epsilon_t^T) = \Sigma$ and $\mathbb{E}(\epsilon_t \epsilon_{t+\tau}^T) = 0$, for $\tau \neq 0$. The VAR process is assumed to be stable and stationary [79], while the noise covariance matrix Σ is assumed to be positive definite with bounded largest eigenvalue, i.e., $\Lambda_{\min}(\Sigma) > 0$ and $\Lambda_{\max}(\Sigma) < \infty$.

In the current context, the parameters $\{A_k\}$ are assumed to be structured, in the sense of having low values according to a suitable norm $R(\cdot)$. We consider a general setting where *any* norm can be applied to the rows $A_k(i,:) \in \mathbb{R}^p$ of A_k, allowing the possibility of different norms being applies to different rows of A_k, and different norms for different parameter matrices $A_k, k = 1, \ldots, d$. Choosing L_1-norm $\|A_k(i,:)\|_1$ for all rows and all parameter matrices is a simple special case of our setting. We discuss certain other choices in Section 4.2.1, and discuss related results in Section 4.4. In order to estimate the parameters, one can consider the standard regularized least-squares estimator. Unfortunately, the samples x_t are not independent, having strong dependence across time and correlated across dimensions. As a result, existing results from the rich literature on regularized estimators for structured problems [117, 118, 119] cannot be directly applied to get sample complexities and estimation error bounds in VAR models.

4.2 Structured VAR Model

In this section we formulate structured VAR estimation problem and discuss its properties, which are essential in characterizing sample complexity and error bounds.

4.2.1 Regularized Estimator

To estimate the parameters of the VAR model, we transform the model in (4.1) into the form suitable for regularized estimator. Specifically, let $(x_0, x_1, \ldots, x_T)$ denote the $T+1$ samples generated by the stable VAR model in (4.1), then stacking them together we obtain

$$\begin{bmatrix} x_d^T \\ x_{d+1}^T \\ \vdots \\ x_T^T \end{bmatrix} = \begin{bmatrix} x_{d-1}^T & x_{d-2}^T & \dots & x_0^T \\ x_d^T & x_{d-1}^T & \dots & x_1^T \\ \vdots & \vdots & \ddots & \vdots \\ x_{T-1}^T & x_{T-2}^T & \dots & x_{T-d}^T \end{bmatrix} \begin{bmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_d^T \end{bmatrix} + \begin{bmatrix} \epsilon_d^T \\ \epsilon_{d+1}^T \\ \vdots \\ \epsilon_T^T \end{bmatrix}$$

which can also be compactly written as

$$Y = XB + E, \tag{4.2}$$

where $Y \in \mathbb{R}^{N\times p}$, $X \in \mathbb{R}^{N\times dp}$, $B \in \mathbb{R}^{dp\times p}$, and $E \in \mathbb{R}^{N\times p}$ for $N = T-d+1$. Vectorizing (column-wise) each matrix in (4.2), we get

$$\begin{aligned} \mathrm{vec}(Y) &= (I_{p\times p} \otimes X)\mathrm{vec}(B) + \mathrm{vec}(E) \\ \mathbf{y} &= Z\boldsymbol{\beta} + \boldsymbol{\epsilon}, \end{aligned}$$

where $\mathbf{y} \in \mathbb{R}^{Np}$, $Z = (I_{p\times p} \otimes X) \in \mathbb{R}^{Np\times dp^2}$, $\boldsymbol{\beta} \in \mathbb{R}^{dp^2}$, $\boldsymbol{\epsilon} \in \mathbb{R}^{Np}$, and $\otimes$ is the Kronecker product. The covariance matrix of the noise $\boldsymbol{\epsilon}$ is now $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] = \Sigma \otimes I_{N\times N}$. Consequently, the regularized estimator takes the form

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{dp^2}}{\mathrm{argmin}} \frac{1}{N}||\mathbf{y} - Z\boldsymbol{\beta}||_2^2 + \lambda_N R(\boldsymbol{\beta}), \tag{4.3}$$

where $R(\boldsymbol{\beta})$ can be any vector norm, separable along the rows of matrices A_k. Specifically, if we denote $\boldsymbol{\beta} = [\beta_1^T \dots \beta_p^T]^T$ and $A_k(i,:)$ as the row of matrix A_k for $k = 1, \dots, d$, then our assumption is equivalent to

$$R(\boldsymbol{\beta}) = \sum_{i=1}^{p} R\big(\beta_i\big) = \sum_{i=1}^{p} R\left(\Big[A_1(i,:)^T \dots A_d(i,:)^T\Big]^T\right). \tag{4.4}$$

To reduce clutter and without loss of generality, we assume the norm $R(\cdot)$ to be the same for each row i. Since the analysis decouples across rows, it is straightforward to extend our analysis to the case when a different norm is used for each row of A_k, e.g., L_1 for row one, L_2 for row two, K-support norm [120] for row three, etc. Observe that within a row, the norm need not be decomposable across columns.

Observe that the estimation problem (4.3) exhibits strong dependence between the samples $(x_0, x_1, \dots, x_T)$, violating the i.i.d. assumption on the data. In particular, this leads to the correlations between the rows and columns of matrix X (and consequently of Z). To deal with such dependencies, following [106], we utilize the spectral representation of the autocovariance of VAR models to control the dependencies in matrix X.

4.2.2 Stability of VAR Model

Since VAR models are (linear) dynamical systems, for the analysis we need to establish conditions under which the VAR model (4.1) is stable, i.e., the time-series process does not diverge over time. For understanding stability, it is convenient to rewrite VAR model of order d in (4.1) as an equivalent VAR model of order 1

$$\begin{bmatrix} x_t \\ x_{t-1} \\ \vdots \\ x_{t-(d-1)} \end{bmatrix} = \underbrace{\begin{bmatrix} A_1 & A_2 & \dots & A_{d-1} & A_d \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} x_{t-1} \\ x_{t-2} \\ \vdots \\ x_{t-d} \end{bmatrix} + \begin{bmatrix} \epsilon_t \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.5}$$

where $\mathbf{A} \in \mathbb{R}^{dp \times dp}$. Therefore, VAR process is stable if all the eigenvalues of $\mathbf{A}$ satisfy $\det(\lambda I_{dp\times dp} - \mathbf{A}) = 0$ for $\lambda \in \mathbb{C}$, $|\lambda| < 1$. Equivalently, if expressed in terms of original parameters A_k, stability is satisfied if $\det(I - \sum_{k=1}^{d} A_k \frac{1}{\lambda^k}) = 0$ (see Appendix 4.A for more details).

4.2.3 Properties of Data Matrix X

In what follows, we analyze the covariance structure of matrix X in (4.2) using spectral properties of VAR model (see Appendix 4.B for additional details). The results will then be used in establishing the high probability bounds for the estimation guarantees in problem (4.3).

Define any row of X as $X_{i,:} \in \mathbb{R}^{dp}$, $1 \leq i \leq N$. Since we assumed that $\epsilon_t \sim \mathcal{N}(0, \Sigma)$, it follows that each row is distributed as $X_{i,:} \sim \mathcal{N}(0, C_{\mathsf{X}})$, where the covariance matrix

$C_{\mathsf{X}} \in \mathbb{R}^{dp\times dp}$ is the same for all i

$$C_{\mathsf{X}} = \begin{bmatrix} \Gamma(0) & \Gamma(1) & \dots & \Gamma(d-1) \\ \Gamma(1)^T & \Gamma(0) & \dots & \Gamma(d-2) \\ \vdots & \vdots & \ddots & \vdots \\ \Gamma(d-1)^T & \Gamma(d-2)^T & \dots & \Gamma(0) \end{bmatrix}, \tag{4.6}$$

where $\Gamma(h) = \mathbb{E}(x_t x_{t+h}^T) \in \mathbb{R}^{p\times p}$. It turns out that since C_{X} is a block-Toeplitz matrix, its eigenvalues can be bounded as (see [121])

$$\inf_{\substack{1\le j\le p\\ \omega\in[0,2\pi]}} \Lambda_j[\rho(\omega)] \le \Lambda_k[C_{\mathsf{X}}] \le \sup_{\substack{1\le j\le p\\ \omega\in[0,2\pi]}} \Lambda_j[\rho(\omega)], \tag{4.7}$$
$$\scriptstyle 1\le k\le dp$$

where $\Lambda_k[\cdot]$ denotes the k-th eigenvalue of a matrix and for $\rho(\omega) = \sum_{h=-\infty}^{\infty} \Gamma(h) e^{-hi\omega}$, where $i = \sqrt{-1}$ and $\omega \in [0, 2\pi]$. $\rho(\omega)$ is the spectral density, i.e., a Fourier transform of the autocovariance matrix $\Gamma(h)$. The advantage of utilizing spectral density is that it has a closed form expression (see Section 9.4 of [122])

$$\rho(\omega) = \left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1} \Sigma \left[\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1}\right]^{*},$$

where $*$ denotes a Hermitian of a matrix. Therefore, from (4.7) we can establish the following lower bound

$$\Lambda_{\min}[C_{\mathsf{X}}] \ge \Lambda_{\min}(\Sigma)/\Lambda_{\max}(\mathcal{A}) = \mathcal{L}, \tag{4.8}$$

where we defined $\Lambda_{\max}(\mathcal{A}) = \max\limits_{\omega\in[0,2\pi]} \Lambda_{\max}(\mathcal{A}(\omega))$ for

$$\mathcal{A}(\omega) = \left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right), \tag{4.9}$$

see Appendix 4.B.1 for additional details.

In establishing high probability bounds we will also need information about a vector $q = Xa \in \mathbb{R}^N$ for any $a \in \mathbb{R}^{dp}$, $\|a\|_2 = 1$. Since each element $X_{i,:}^T a \sim \mathcal{N}(0, a^T C_{\mathsf{X}} a)$, it follows that $q \sim \mathcal{N}(0, Q_a)$ with a covariance matrix $Q_a \in \mathbb{R}^{N\times N}$. It can be shown (see Appendix 4.B.3 for more details) that Q_a can be written as

$$Q_a = (I \otimes a^T) C_{\mathcal{U}} (I \otimes a), \tag{4.10}$$

where $C_{\mathcal{U}} = \mathbb{E}(\mathcal{U}\mathcal{U}^T)$ for $\mathcal{U} = \begin{bmatrix} X_{1,:}^T & \dots & X_{N,:}^T \end{bmatrix}^T \in \mathbb{R}^{Ndp}$ which is obtained from matrix X by stacking all the rows in a single vector, i.e, $\mathcal{U} = \text{vec}(X^T)$. In order to bound eigenvalues of $C_{\mathcal{U}}$ (and consequently of Q_a), observe that $\mathcal{U}$ can be viewed as a vector obtained by stacking N outputs from VAR model in (4.5). Similarly as in (4.7), if we denote the spectral density of the VAR process in (4.5) as $\rho_{\mathsf{X}}(\omega) = \sum_{h=-\infty}^{\infty} \Gamma_{\mathsf{X}}(h) e^{-hi\omega}$, $\omega \in [0, 2\pi]$, where $\Gamma_{\mathsf{X}}(h) = \mathbb{E}[X_{j,:}X_{j+h,:}^T] \in \mathbb{R}^{dp\times dp}$, then we can write

$$\inf_{\substack{1\le l\le dp\\ \omega\in[0,2\pi]}} \Lambda_l[\rho_{\mathsf{X}}(\omega)] \le \Lambda_k[C_{\mathcal{U}}] \le \sup_{\substack{1\le l\le dp\\ \omega\in[0,2\pi]}} \Lambda_l[\rho_{\mathsf{X}}(\omega)].$$

with $1\le k\le Ndp$ under $\Lambda_k[C_{\mathcal{U}}]$.

The closed form expression of spectral density is

$$\rho_{\mathsf{X}}(\omega) = \left(I - \mathbf{A}e^{-i\omega}\right)^{-1} \Sigma_{\mathcal{E}} \left[\left(I - \mathbf{A}e^{-i\omega}\right)^{-1}\right]^*,$$

where $\Sigma_{\mathcal{E}}$ is the covariance matrix of a noise vector and $\mathbf{A}$ are as defined in expression (4.5). Thus, an upper bound on $C_{\mathcal{U}}$ can be obtained as $\Lambda_{\max}[C_{\mathcal{U}}] \le \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})}$, where we defined $\Lambda_{\min}(\boldsymbol{\mathcal{A}}) = \min_{\omega\in[0,2\pi]} \Lambda_{\min}(\boldsymbol{\mathcal{A}}(\omega))$ for

$$\boldsymbol{\mathcal{A}}(\omega) = \left(I - \mathbf{A}^T e^{i\omega}\right)\left(I - \mathbf{A}e^{-i\omega}\right). \tag{4.11}$$

Referring back to covariance matrix Q_a in (4.10), we get

$$\Lambda_{\max}[Q_a] \le \Lambda_{\max}(\Sigma)/\Lambda_{\min}(\boldsymbol{\mathcal{A}}) = \mathcal{M}. \tag{4.12}$$

We note that for a general VAR model, there might not exist closed-form expressions for $\Lambda_{\max}(\mathcal{A})$ and $\Lambda_{\min}(\boldsymbol{\mathcal{A}})$. However, for some special cases there are results establishing the bounds on these quantities (e.g., see Proposition 2.2 in [106]).

4.3 Regularized Estimation Guarantees

Denote by $\Delta = \hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*$ the error between the solution of optimization problem (4.3) and $\boldsymbol{\beta}^*$, the true value of the parameter. The focus of our work is to determine conditions under which the optimization problem in (4.3) has guarantees on the accuracy of the obtained solution, i.e., the error term is bounded: $||\Delta||_2 \le \delta$ for some known δ. To establish such conditions, we utilize the framework of [109]. Specifically, estimation error analysis is based on the following known results adapted to our settings. The first one characterizes the restricted error set Ω_E, where the error Δ belongs.

Lemma 9 *Assume that*

$$\lambda_N \geq rR^* \left[\frac{1}{N} Z^T \boldsymbol{\epsilon}\right], \tag{4.13}$$

for some constant $r > 1$*, where* $R^*\left[\frac{1}{N}Z^T\boldsymbol{\epsilon}\right]$ *is a dual form of the vector norm* $R(\cdot)$*, which is defined as* $R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}] = \sup_{R(U)\leq 1} \left\langle \frac{1}{N}Z^T\boldsymbol{\epsilon}, U\right\rangle$*, for* $U \in \mathbb{R}^{dp^2}$*, where* $U = [u_1^T, u_2^T, \dots, u_p^T]^T$ *and* $u_i \in \mathbb{R}^{dp}$*. Then the error vector* $\|\Delta\|_2$ *belongs to the set*

$$\Omega_E = \left\{ \Delta \in \mathbb{R}^{dp^2} \middle| R(\boldsymbol{\beta}^* + \Delta) \leq R(\boldsymbol{\beta}^*) + \frac{1}{r} R(\Delta) \right\}. \tag{4.14}$$

The second condition in [109] establishes the upper bound on the estimation error.

Lemma 10 *Assume that the restricted eigenvalue (RE) condition holds*

$$\frac{||Z\Delta||_2}{||\Delta||_2} \geq \sqrt{\kappa N}, \tag{4.15}$$

for $\Delta \in \text{cone}(\Omega_E)$ *and some constant* $\kappa > 0$*, where* $\text{cone}(\Omega_E)$ *is a cone of the error set, then*

$$||\Delta||_2 \leq \frac{1+r}{r} \frac{\lambda_N}{\kappa} \Psi(\text{cone}(\Omega_E)), \tag{4.16}$$

where $\Psi(\text{cone}(\Omega_E))$ *is a norm compatibility constant, which is defined as* $\Psi(\text{cone}(\Omega_E)) = \sup_{U \in \text{cone}(\Omega_E)} \frac{R(U)}{||U||_2}$.

Note that the above error bound is deterministic, i.e., if (4.13) and (4.15) hold, then the error satisfies the upper bound in (4.16). However, the results are defined in terms of the quantities, involving Z and $\boldsymbol{\epsilon}$, which are random. Therefore, in the following we establish high probability bounds on the regularization parameter in (4.13) and RE condition in (4.15).

4.3.1 High Probability Bounds

In this Section we present the main results of our work, followed by the discussion on their properties and illustrating some special cases based on popular Lasso and Group Lasso regularization norms. In Section 4.3.4 we will present the main ideas of our proof technique, with all the details delegated to the Appendices 4.C and 4.D.

To establish lower bound on the regularization parameter λ_N, we derive an upper bound on $R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}] \leq \alpha$, for some $\alpha > 0$, which will establish the required relationship $\lambda_N \geq \alpha \geq R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}]$.

Theorem 11 *Let $\Omega_R = \{u \in \mathbb{R}^{dp} | R(u) \leq 1\}$, and define $w(\Omega_R) = \mathbb{E}[\sup_{u\in\Omega_R} \langle g, u\rangle]$ to be a Gaussian width of set Ω_R for $g \sim \mathcal{N}(0, I)$. For any $\epsilon_1 > 0$ and $\epsilon_2 > 0$ with probability at least $1 - c\exp(-\min(\epsilon_2^2, \epsilon_1) + \log(p))$ we can establish that*

$$R^*\left[\frac{1}{N}Z^T\boldsymbol{\epsilon}\right] \leq \left(c_2(1+\epsilon_2)\frac{w(\Omega_R)}{\sqrt{N}} + c_1(1+\epsilon_1)\frac{w^2(\Omega_R)}{N^2}\right)$$

where c, c_1 and c_2 are positive constants.

To establish restricted eigenvalue condition, we will show that $\inf_{\Delta\in\text{cone}(\Omega_E)} \frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \geq \nu$, for some $\nu > 0$ and then set $\sqrt{\kappa N} = \nu$.

Theorem 12 *Let $\Theta = \text{cone}(\Omega_{E_j}) \cap S^{dp-1}$, where S^{dp-1} is a unit sphere. The error set Ω_{E_j} is defined as $\Omega_{E_j} = \left\{\Delta_j \in \mathbb{R}^{dp}\middle| R(\beta_j^* + \Delta_j) \leq R(\beta_j^*) + \frac{1}{r}R(\Delta_j)\right\}$, for $r > 1$, $j = 1, \ldots, p$, and $\Delta = [\Delta_1^T, \ldots, \Delta_p^T]^T$, for Δ_j is of size $dp \times 1$, and $\boldsymbol{\beta}^* = [\beta_1^{*T} \ldots \beta_p^{*T}]^T$, for $\beta_j^* \in \mathbb{R}^{dp}$. The set Ω_{E_j} is a part of the decomposition in $\Omega_E = \Omega_{E_1} \times \cdots \times \Omega_{E_p}$ due to the assumption on the row-wise separability of norm $R(\cdot)$ in (4.4). Also define $w(\Theta) = \mathbb{E}[\sup_{u\in\Theta} \langle g, u\rangle]$ to be a Gaussian width of set Θ for $g \sim \mathcal{N}(0, I)$ and $u \in \mathbb{R}^{dp}$. Then with probability at least $1 - c_1\exp(-c_2\eta^2 + \log(p))$, for any $\eta > 0$*

$$\inf_{\Delta\in\text{cone}(\Omega_E)} \frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \geq \nu,$$

where $\nu = \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta$ and c, c_1, c_2 are positive constants, and $\mathcal{L}$ and $\mathcal{M}$ are defined in (4.8) and (4.12).

4.3.2 Discussion

From Theorem 12, we can choose $\eta = \frac{1}{2}\sqrt{N\mathcal{L}}$ and set $\sqrt{\kappa N} = \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta$ and since $\sqrt{\kappa N} > 0$ must be satisfied, we can establish a lower bound on the number of samples N

$$\sqrt{N} > \frac{2\sqrt{\mathcal{M}} + cw(\Theta)}{\sqrt{\mathcal{L}}/2} = \mathcal{O}(w(\Theta)). \tag{4.17}$$

Examining this bound and using (4.8) and (4.12), we can conclude that the number of samples needed to satisfy the restricted eigenvalue condition is smaller if $\Lambda_{\min}(\boldsymbol{\mathcal{A}})$ and $\Lambda_{\min}(\Sigma)$ are larger and $\Lambda_{\max}(\mathcal{A})$ and $\Lambda_{\max}(\Sigma)$ are smaller. In turn, this means that matrices $\mathcal{A}$ and $\boldsymbol{\mathcal{A}}$ in (4.9) and (4.11) must be well conditioned and the VAR process is stable, with eigenvalues well inside the unit circle (see Section 4.2.2). Alternatively, we can also understand (4.17) as showing that large values of $\mathcal{M}$ and small values of $\mathcal{L}$ indicate stronger dependency in the data, thus requiring more samples for the RE conditions to hold with high probability.

Analyzing Theorems 11 and 12 we can interpret the established results as follows. As the size and dimensionality N, p and d of the problem increase, we emphasize the scale of the results and use the order notations to denote the constants. Select a number of samples at least $N \geq \mathcal{O}(w^2(\Theta))$ and let the regularization parameter satisfy $\lambda_N \geq \mathcal{O}\left(\frac{w(\Omega_R)}{\sqrt{N}} + \frac{w^2(\Omega_R)}{N^2}\right)$. With high probability then the restricted eigenvalue condition $\frac{||Z\Delta||_2}{||\Delta||_2} \geq \sqrt{\kappa N}$ for $\Delta \in \text{cone}(\Omega_E)$ holds, so that $\kappa = \mathcal{O}(1)$ is a positive constant. Moreover, the norm of the estimation error in optimization problem (4.3) is bounded by $\|\Delta\|_2 \leq \mathcal{O}\left(\frac{w(\Omega_R)}{\sqrt{N}} + \frac{w^2(\Omega_R)}{N^2}\right)\Psi(\text{cone}(\Omega_{E_j}))$. Note that the norm compatibility constant $\Psi(\text{cone}(\Omega_{E_j}))$ is assumed to be the same for all $j = 1, \ldots, p$, which follows from our assumption in (4.4).

Consider now Theorem 11 and the bound on the regularization parameter $\lambda_N \geq \mathcal{O}\left(\frac{w(\Omega_R)}{\sqrt{N}} + \frac{w^2(\Omega_R)}{N^2}\right)$. As the dimensionality of the problem p and d grows and the number of samples N increases, the first term $\frac{w(\Omega_R)}{\sqrt{N}}$ will dominate the second one $\frac{w^2(\Omega_R)}{N^2}$. This can be seen by computing N for which the two terms become equal $\frac{w(\Omega_R)}{\sqrt{N}} = \frac{w^2(\Omega_R)}{N^2}$, which happens at $N = w^{\frac{2}{3}}(\Omega_R) < w(\Omega_R)$. Therefore, we can rewrite our results as follows: once the restricted eigenvalue condition holds and $\lambda_N \geq \mathcal{O}\left(\frac{w(\Omega_R)}{\sqrt{N}}\right)$, the error norm is upper-bounded by $\|\Delta\|_2 \leq \mathcal{O}\left(\frac{w(\Omega_R)}{\sqrt{N}}\right)\Psi(\text{cone}(\Omega_{E_j}))$.

4.3.3 Special Cases

While the presented results are valid for any norm $R(\cdot)$, separable along the rows of A_k, it is instructive to specialize our analysis to a few popular regularization choices, such as L_1 and Group Lasso, Sparse Group Lasso and OWL norms.

Lasso To establish results for L_1 norm, we assume that the parameter $\boldsymbol{\beta}^*$ is s-sparse, which in our case is meant to represent the largest number of non-zero elements in any β_i, $i = 1, \ldots, p$, i.e., the combined i-th rows of each A_k, $k = 1, \ldots, d$. Since L_1 is decomposable, it can be shown that $\Psi(\text{cone}(\Omega_{E_j})) \leq 4\sqrt{s}$. Next, since $\Omega_R = \{u \in \mathbb{R}^{dp} | R(u) \leq 1\}$, then using Lemma 3 in [109] and Gaussian width results in [115], we can establish that $w(\Omega_R) \leq \mathcal{O}(\sqrt{\log(dp)})$. Therefore, based on Theorem 4.3 and the discussion at the end of Section 4.3.2, the bound on the regularization parameter takes the form $\lambda_N \geq \mathcal{O}\left(\sqrt{\log(dp)/N}\right)$. Hence, the estimation error is bounded by $\|\Delta\|_2 \leq \mathcal{O}\left(\sqrt{s\log(dp)/N}\right)$ as long as $N > \mathcal{O}(\log(dp))$.

Group Lasso To establish results for Group norm, we assume that for each $i = 1, \ldots, p$, the vector $\beta_i \in \mathbb{R}^{dp}$ can be partitioned into a set of K disjoint groups, $G = \{G_1, \ldots, G_K\}$, with the size of the largest group $m = \max_k |G_k|$. Group Lasso norm is defined as $\|\boldsymbol{\beta}\|_{\text{GL}} = \sum_{k=1}^{K} \|\beta_{G_k}\|_2$. We assume that the parameter $\boldsymbol{\beta}^*$ is s_G-group-sparse, which means that the largest number of non-zero groups in any β_i, $i = 1, \ldots, p$ is s_G. Since Group norm is decomposable, as was established in [107], it can be shown that $\Psi(\text{cone}(\Omega_{E_j})) \leq 4\sqrt{s_G}$. Similarly as in the Lasso case, using Lemma 3 in [109], we get $w(\Omega_{R_{\text{GL}}}) \leq \mathcal{O}(\sqrt{m + \log(K)})$. The bound on the λ_N takes the form $\lambda_N \geq \mathcal{O}\left(\sqrt{(m + \log(K))/N}\right)$. Combining these derivations, we obtain the bound $\|\Delta\|_2 \leq \mathcal{O}\left(\sqrt{s_G(m + \log(K))/N}\right)$ for $N > \mathcal{O}(m + \log(K))$.

Sparse Group Lasso Similarly as in Section 4.3.3, we assume that we have K disjoint groups of size at most m. The Sparse Group Lasso norm enforces sparsity not only across but also within the groups and is defined as $\|\boldsymbol{\beta}\|_{\text{SGL}} = \alpha\|\boldsymbol{\beta}\|_1 + (1 - \alpha)\sum_{k=1}^{K} \|\beta_{G_k}\|_2$, where $\alpha \in [0, 1]$ is a parameter which regulates a convex combination of Lasso and Group Lasso penalties. Note that since $\|\boldsymbol{\beta}\|_2 \leq \|\boldsymbol{\beta}\|_1$, it follows that $\|\boldsymbol{\beta}\|_{\text{GL}} \leq \|\boldsymbol{\beta}\|_{\text{SGL}}$. As a result, for $\boldsymbol{\beta} \in \Omega_{R_{\text{SGL}}} \Rightarrow \boldsymbol{\beta} \in \Omega_{R_{\text{GL}}}$, so that $\Omega_{R_{\text{SGL}}} \subseteq \Omega_{R_{\text{GL}}}$ and thus $w(\Omega_{R_{\text{SGL}}}) \leq w(\Omega_{R_{\text{GL}}}) \leq \mathcal{O}(\sqrt{m + \log(K)})$, according to Section 4.3.3. Assuming $\boldsymbol{\beta}^*$ is s-sparse and s_G-group-sparse and noting that the norm is decomposable, we get $\Psi(\text{cone}(\Omega_{E_j})) \leq 4(\alpha\sqrt{s} + (1 - \alpha)\sqrt{s_G}))$. Consequently, the error bound is $\|\Delta\|_2 \leq \mathcal{O}\left(\sqrt{(\alpha s + (1 - \alpha)s_G)(m + \log(K))/N}\right)$.

OWL norm Ordered weighted L_1 norm is a recently introduced regularizer and is defined as $\|\boldsymbol{\beta}\|_{\text{owl}} = \sum_{i=1}^{dp} c_i|\beta|_{(i)}$, where $c_1 \geq \ldots \geq c_{dp} \geq 0$ is a predefined non-increasing sequence of weights and $|\beta|_{(1)} \geq \ldots \geq |\beta|_{(dp)}$ is the sequence of absolute values of $\boldsymbol{\beta}$, ranked in decreasing order. In [116] it was shown that $w(\Omega_R) \leq \mathcal{O}(\sqrt{\log(dp)}/\bar{c})$, where $\bar{c}$ is the average of $c_1, \ldots, c_{dp}$ and the norm compatibility constant is $\Psi(\text{cone}(\Omega_{E_j})) \leq 2c_1^2\sqrt{s}/\bar{c}$. Therefore, based on Theorem 4.3, we get $\lambda_N \geq \mathcal{O}\left(\sqrt{\log(dp)/(\bar{c}N)}\right)$ and the estimation error is bounded by $\|\Delta\|_2 \leq \mathcal{O}\left(\frac{2c_1}{\bar{c}}\sqrt{s\log(dp)/(\bar{c}N)}\right)$.

We note that the bound obtained for Lasso and Group Lasso is similar to the bound obtained in [103, 106, 104]. Moreover, this result is also similar to the works, which dealt with independent observations, e.g., [123, 107], with the difference being the constants, reflecting correlation between the samples, as we discussed in Section 4.3.2. The explicit bound for Sparse Group Lasso and OWL is a *novel* aspect of our work for the non-asymptotic recovery guarantees for the VAR estimation problem with norm regularization, being just a simple consequence from our more general framework.

4.3.4 Proof Sketch

In this Section we outline the steps of the proof for Theorem 11 and 12, all the details can be found in Appendix 4.C and 4.D.

Bound on Regularization Parameter Recall that our objective is to establish for $\alpha > 0$ a probabilistic statement that $\lambda_N \geq \alpha \geq R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}] = \sup_{R(U)\leq 1} \left\langle \frac{1}{N}Z^T\boldsymbol{\epsilon}, U\right\rangle$, where $U = [u_1^T, \ldots, u_p^T]^T \in \mathbb{R}^{dp^2}$ for $u_j \in \mathbb{R}^{dp}$ and $\boldsymbol{\epsilon} = \text{vec}(E)$ for E in (4.2). We denote $E_{:,j} \in \mathbb{R}^N$ as a column of noise matrix E and note that since $Z = I_{p\times p} \otimes X$, then using the row-wise separability assumption in (4.4) we can split the overall probability statement into p parts, which are easier to work with. Thus, our objective would be to establish

$$\mathbb{P}\left[\sup_{R(u_j)\leq r_j} \frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \alpha_j\right] \geq \pi_j, \tag{4.18}$$

for $j = 1, \ldots, p$, where $\sum_{j=1}^p \alpha_j = \alpha$ and $\sum_{j=1}^p r_j = 1$.

The overall strategy is to first show that the random variable $\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$ has sub-exponential tails. Based on the generic chaining argument, we then use Theorem 1.2.7 from [113] and bound the expectation $\mathbb{E}\left[\sup_{R(u_j)\leq r_j} \frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right]$. Finally, using Theorem 1.2.9 in [113] we establish the high probability bound on concentration of $\sup_{R(u_j)\leq r_j} \frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$ around its mean, i.e., derive the bound in (4.18).

We note that the main difficulty of working with the term $\left\langle X^T E_{:,j}, u_j\right\rangle$ is the complicated dependency between X and $E_{:,j}$, which is due to the VAR generation process in (4.2). However, if we write $\left\langle X^T E_{:,j}, u_j\right\rangle = \sum_{i=1}^N E_{i,j},(X_{i,:}u_j) = \sum_{i=1}^N m_i$, where $m_i = E_{i,j}(X_{i,:}u_j)$ and we can interpret this as a summation over martingale difference sequence [79]. This can be easily proven by showing $\mathbb{E}(m_i|m_1,\dots,m_{i-1}) = 0$. The latter is true since in $m_i = E_{i,j}(X_{i,:}u_j)$ the terms $E_{i,j}$ and $X_{i,:}u_j$ are independent since ϵ_{d+i} is independent from x_{d-k+i} for $0 \leq i \leq T-d$ and $1 \leq k \leq d$ (see (4.1)).

To show that $\sum_{i=1}^N E_{i,j},(X_{:,i}u_j)$ has sub-exponential tails, recall that since ϵ_t in (4.1) is Gaussian, $E_{i,j}$ and $X_{i,:}u_j$ are independent Gaussian random variables, whose product has sub-exponential tails. Moreover, the sum over sub-exponential martingale difference sequence can be shown to be itself sub-exponential using [124], based on Bernstein-type inequality [125].

Restricted Eigenvalue Condition To show $\frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \geq 0$ for all $\Delta \in \text{cone}(\Omega_E)$, similarly as before, we split the problem into p parts by using row-wise separability assumption of the norm in (4.4). In particular, denote $\Delta = [\Delta_1^T,\dots,\Delta_p^T]^T$, where Δ_j is $dp \times 1$, then we can represent the original set Ω_E as a Cartesian product of subsets Ω_{E_j}, i.e., $\Omega_E = \Omega_{E_1} \times \cdots \times \Omega_{E_p}$, implying that $\text{cone}(\Omega_E) = \text{cone}(\Omega_{E_1}) \times \cdots \times \text{cone}(\Omega_{E_p})$. Therefore, our objective would be to establish

$$\mathbb{P}\left[\inf_{u_j\in\Theta_j} ||Xu_j||_2 \geq \nu_j\right] \geq \pi_j, \tag{4.19}$$

for each $j = 1,\dots,p$, where $\Theta = \text{cone}(\Omega_{E_j}) \cap S^{dp-1}$ and we defined $u_j = \frac{\Delta_j}{||\Delta_j||_2}$, since it will be easier to operate with unit-norm vectors. In the following, to reduce clutter, we drop the index j from the notations.

The overall strategy is to first show that $\|Xu\|_2 - \mathbb{E}(\|Xu\|_2)$ is a sub-Gaussian random variable. Then, using generic chaining argument in [113], specifically Theorem 2.1.5, we bound $\mathbb{E}\left(\inf_{u\in\Theta} ||Xu||_2\right)$. Finally, based on Lemma 2.1.3 in [113] we establish the concentration inequality on $\inf_{u\in\Theta}||Xu||_2$ around its mean, i.e., derive the bound in (4.19).

4.4 Experimental Results

In this Section we present the experiments on simulated and real data to demonstrate the obtained theoretical results. In particular, for L_1 and Group L_1, Sparse Group L_1 and OWL we investigate how error norm $\|\Delta\|_2$ and regularization parameter λ_N scale as the problem size p and N change. Moreover, using flight data we also compare the performance of the regularizers in real world scenario.

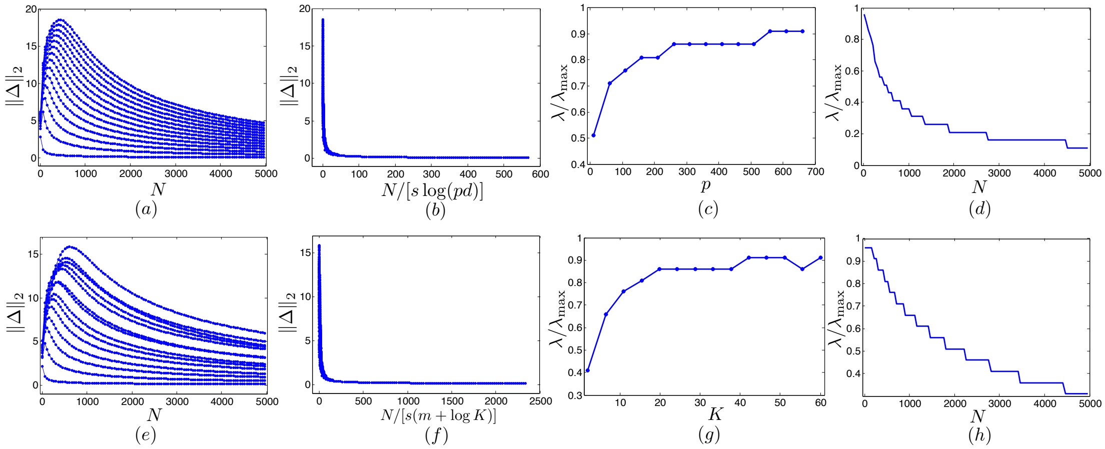
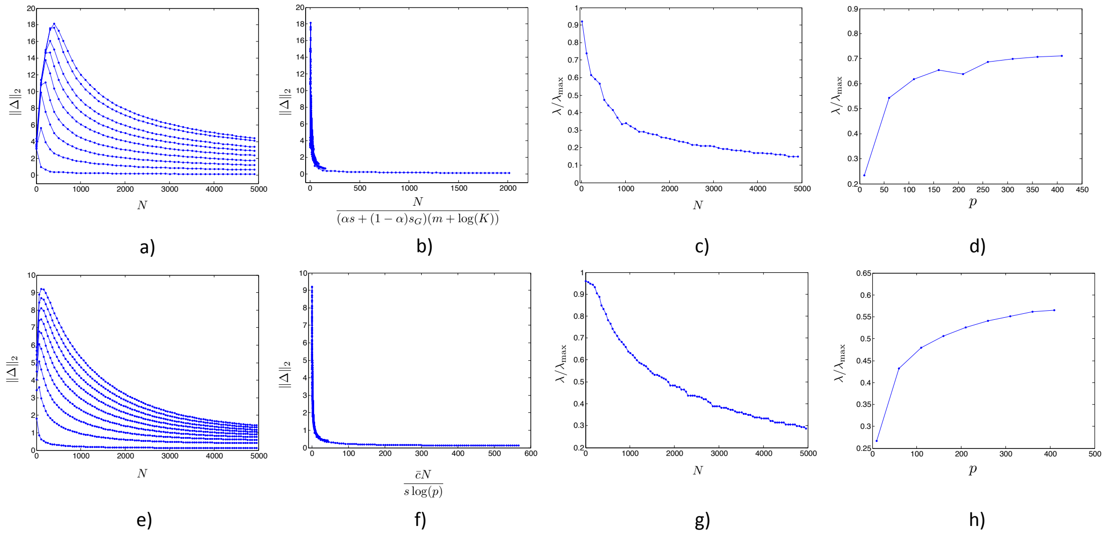


Figure 4.1: Results for estimating parameters of a stable first order sparse VAR (top row) and group sparse VAR (bottom row). Problem dimensions: $p \in [10, 600]$, $N \in [10, 5000]$, $\frac{\lambda_N}{\lambda_{max}} \in [0, 1]$, $K \in [2, 60]$ and $d = 1$. Figures (a) and (e) show dependency of errors on sample size for different p; in Figure (b) the N is scaled by $(s \log p)$ and plotted against $\|\Delta\|_2$ to show that errors scale as $(s \log p)/N$; in (f) the graph is similar to (b) but for group sparse VAR; in (c) and (g) we show dependency of λ_N on p (or number of groups K in (g)) for fixed sample size N; finally, Figures (d) and (h) display the dependency of λ_N on N for fixed p.



a) b) c) d)

e) f) g) h)

Figure 4.2: Results for estimating parameters of a stable first order Sparse Group Lasso VAR (top row) and OWL-regularized VAR (bottom row). Problem dimensions for Sparse Group Lasso : $p \in [10, 410]$, $N \in [10, 5000]$, $\frac{\lambda_N}{\lambda_{max}} \in [0, 1]$, $K \in [2, 60]$ and $d = 1$. Problem dimensions for OWL: $p \in [10, 410]$, $N \in [10, 5000]$, $\frac{\lambda_N}{\lambda_{max}} \in [0, 1]$, $s \in [4, 260]$ and $d = 1$. All results are shown after averaging across 50 runs.

4.4.1 Synthetic Data

Using synthetically generated datasets we evaluate the obtained theoretical bounds for estimation VAR under Lasso, Sparse Group Lasso, OWL and Group Lasso regularizations.

Lasso To evaluate the estimation problem with L_1 norm, we simulated a first-order VAR process for different values of $p \in [10, 600]$, $s \in [4, 260]$, and $N \in [10, 5000]$. Regularization parameter was varied in the range $\lambda_N \in (0, \lambda_{\max})$, where $\lambda_{\max}$ is the largest parameter, for which estimation problem (4.3) produces a zero solution. All the results are shown after averaging across 50 runs.

The results for Lasso are shown in the top row of Figure 4.1. In particular, in Figure 4.1.a we show $\|\Delta\|_2$ for different p and N for fixed λ_N. When N is small, the estimation error is large and the results cannot be trusted. However, once $N \geq \mathcal{O}(w^2(\Theta))$, the RE condition in Lemma 10 is satisfied and we see a fast decrease of errors for all p's.

In Figure 4.1.b we plot $\|\Delta\|_2$ against rescaled sample size $\frac{N}{s\log(pd)}$. The errors are now closely aligned, confirming results of Section 4.3.3, i.e, $\|\Delta\|_2 \leq \mathcal{O}\left(\sqrt{(s\log(pd))/N}\right)$.

Finally, in Figures 4.1.c and 4.1.d we show the dependence of optimal λ_N (for fixed N and p, we picked λ_N achieving the smallest estimation error) on N and p. It can be seen that as p increases, λ_N grows (for fixed N) at the rate similar to $\sqrt{\log p}$. On the other hand, as N increases, the selected λ_N decreases (for fixed p) at the rate similar to $1/\sqrt{N}$.

Sparse Group Lasso To evaluate the estimation problem with Sparse Group Lasso norm, we constructed first-order VAR process for the following set of problem sizes $p \in [10, 400]$, $s \in [10, 200]$, $s_G \in [2, 20]$ and $N \in [10, 5000]$. The parameter α was set to 0.5. Results are shown in Figure 4.2, top row. Similarly as in main paper, we can see that the errors are scaled by $\frac{N}{(\alpha s+(1-\alpha)s_G)(m+\log(K))}$. Moreover, the λ_N parameter is decreasing when number of samples N increases. On the other hand, as the problem dimension p increases, the selected λ_N grows at the rate similar to $\sqrt{\log p}$.

OWL To test the VAR estimation problem under OWL norm we constructed a first-order VAR process with $p \in [10, 410]$, $s \in [4, 260]$ and $N \in [10, 5000]$. The vector of weights c was set to be a monotonically decreasing sequence of numbers in the range $[1, 0)$. Figure 4.2, bottom row, shows the results. It can be seen from Figure 4.2-f that when the errors are plotted against $\frac{\bar{c}N}{s\log(p)}$, they become tightly aligned, confirming the bounds established in Section 3.3.4 in the main paper for the error norm. As shown in Figure 4.2-g,h the selected regularization parameter λ_N grows with the problem dimension p and decreases with the number of samples N

Group Lasso For Group Lasso the sparsity in rows of A_1 was generated in groups, whose number varied as $K \in [2, 60]$. We set the largest number of non-zero groups in any row as $s_G \in [2, 22]$. Results are shown in the bottom row of Figure 4.1, which have similar flavor as in Lasso case. The difference can be seen in Figure 4.1.f, where a close alignment of errors occurs when N is now scaled as $\frac{N}{s_G(m+\log(K))}$. Moreover, the selected regularization parameter λ increases with the number of groups K and decreases with N.

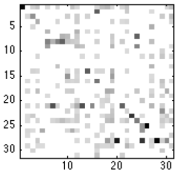
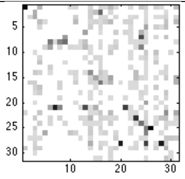
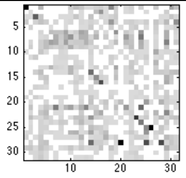
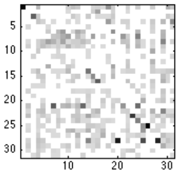
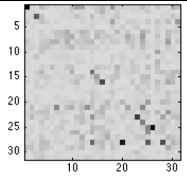
Lasso	OWL	Group Lasso	Sparse Group Lasso	Ridge
32.3(6.5)	32.2(6.6)	32.7(6.5)	32.2(6.4)	33.5(6.1)
32.7(7.9)	44.5(15.6)	75.3(8.4)	38.4(9.6)	99.9(0.2)
				

Table 4.1: Mean squared error (row 2) of the five methods used in fitting VAR model, evaluated on aviation dataset (MSE is computed using one-step-ahead prediction errors). Row 3 shows the average number of non-zeros (as a percentage of total number of elements) in the VAR matrix. The last row shows a typical sparsity pattern in A_1 for each method (darker dots - stronger dependencies, lighter dots - weaker dependencies). The values in parenthesis denote one standard deviation after averaging the results over 300 flights.

4.4.2 Real Data

We have also performed evaluation tests on real data to compare the accuracy of the VAR estimation using various penalized formulations based on five norms: L_1, OWL, Group, Sparse Group and Ridge (square of L_2). Although $\|\cdot\|_2^2$ is not a norm, we included its results for reference purposes as it is frequently used in practice. In terms of data, we used the NASA flight dataset from [42], consisting of over 100,000 flights, each having a record of about 250 parameters, sampled at 1 Hz. For our test, we selected 300 flights and picked 31 parameters most suitable for the prediction task (shown in Table 4.2) and focused on the landing part of the trajectory (duration approximately 15 minutes). For each flight we separately fitted a first-order VAR model using five approaches and performed 5-fold cross validation to select λ, achieving smallest prediction error. For Sparse Group we set $\alpha = 0.5$, while for OWL the weights $c_1, \dots, c_p$ were set as a monotonically decreasing sequence. Table 4.1 shows the results after averaging across 300 flights.

From the table we can see that the considered problem exhibits a sparse structure since all the methods detected similar patterns in matrix A_1. In particular, the analysis of such patterns revealed a meaningful relationship among the flight parameters (darker dots), e.g., normal acceleration had high dependency on vertical speed and

1	Altitude
2	Corrected angle of attack
3	Brake temperature
4	Computed airspeed
5	Drift angle
6	Engine temperature
7	Low rotor speed
8	High rotor speed
9	Engine oil pressure
10	Engine oil quantity
11	Engine oil temperature
12	Engine pre-cooler outlet temperature
13	Fuel mass flow rate
14	Lateral acceleration
15	Longitudinal acceleration
16	Normal acceleration
17	Glide slope deviation
18	Ground speed
19	Localization deviation
20	Magnetic heading
21	Burner pressure
22	Pitch angle
23	Roll angle
24	HPC exit temperature
25	Angle magnitude
26	Angle true
27	Total fuel quantity
28	True heading
29	Vertical speed
30	True airspeed
31	MACH

Table 4.2: 31 features selected for structured VAR estimation on real flight data.

angle-of-attack, the altitude had mainly dependency with fuel quantity, vertical speed with aircraft nose pitch angle, etc. The results also showed that the sparse regularization helps in recovering more accurate and parsimonious models as is evident by comparing performance of Ridge regression with other methods. Moreover, while all the four Lasso-based approaches performed similar to each other, their sparsity levels were different, with Lasso producing the sparsest solutions. As was also expected, Group Lasso had larger number of non-zeros since it did not enforce sparsity within the groups, as compared to the sparse version of this norm.

Appendix

4.A Stability of VAR Model

A VAR process is stable if all the eigenvalues of $\mathbf{A}$, defined in (4.5), are smaller than 1, i.e., eigenvalues of $\mathbf{A}$ must satisfy $\det(\lambda I_{dp\times dp} - \mathbf{A}) = 0$ for $\lambda \in \mathbb{C}$, $|\lambda| < 1$, $|\lambda| \neq 0$. Specifically, write

$$\lambda I_{dp\times dp} - \mathbf{A} = \begin{bmatrix} I\lambda & 0 & \dots & 0 & 0 \\ 0 & I\lambda & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & I\lambda \end{bmatrix} - \begin{bmatrix} A_1 & A_2 & \dots & A_{d-1} & A_d \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix}$$
$$= \begin{bmatrix} I\lambda - A_1 & -A_2 & \dots & -A_{d-1} & -A_d \\ -I & I\lambda & \dots & 0 & 0 \\ 0 & -I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -I & I\lambda \end{bmatrix}.$$

Now multiply last (d-th) block-column by $\frac{1}{\lambda}$ and add to $(d-1)$-st block-column. Next, multiply the result in $(d-1)$-st block-column by $\frac{1}{\lambda}$ and add to $(d-2)$-nd block-column. Continuing in this manner, we will arrive at

$$Q = \begin{bmatrix} \lambda I_{p\times p} - A_1 - \frac{1}{\lambda}A_2 - \dots - \frac{1}{\lambda^{d-1}}A_d & M \\ 0 & \lambda I_{p(d-1)\times p(d-1)} \end{bmatrix},$$

where matrix $M \in \mathbb{R}^{p\times p(d-1)}$ denotes the result of some of the column operations. Since such column operations leave the matrix determinant unchanged, we have

$$\begin{aligned}\det(\lambda I_{dp\times dp} - \mathbf{A}) = \det(Q) &= \det(\lambda I_{p\times p} - A_1 - \frac{1}{\lambda}A_2 - \ldots - \frac{1}{\lambda^{d-1}}A_d) \cdot \det(\lambda I_{p(d-1)\times p(d-1)}) \\ &= \det(I_{p\times p} - \frac{1}{\lambda}A_1 - \frac{1}{\lambda^2}A_2 - \ldots - \frac{1}{\lambda^d}A_d) \cdot \lambda^{pd}.\end{aligned}$$

Therefore, stability of VAR model in (4.5) requires $\det(I - \sum_{k=1}^{d} A_k \frac{1}{\lambda^k}) = 0$ to be satisfied for $|\lambda| < 1$, $|\lambda| \neq 0$. Equivalently, $\det(I - \sum_{k=1}^{d} A_k z^k) = 0$ must be satisfied for $z \in \mathbb{C}$, $|z| > 1$, or $\det(I - \sum_{k=1}^{d} A_k z^k) \neq 0$ must hold for $|z| \leq 1$. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

4.B Properties of Data Matrix X

In this Section we provide additional details about the covariance structure of VAR matrix X as was originally presented in Section 4.2.3. Recall that our VAR process is defined as

$$x_t = A_1 x_{t-1} + \ldots + A_d x_{t-d} + \epsilon_t, \quad t = 0, \pm 1, \pm 2, \ldots, \tag{4.20}$$

where noise ϵ_t follows a Gaussian distribution, i.e., $\epsilon_t \sim \mathcal{N}(0, \Sigma)$, moreover, the distribution of x_t is a zero-mean Gaussian, i.e., $x_t \sim \mathcal{N}(0, \Gamma(0))$, where $\Gamma(h) = \mathbb{E}(x_t x_{t+h}^T)$.

Now consider the noise and data matrices from the formulation (4.2)

$$X = \begin{bmatrix} x_{d-1}^T & x_{d-2}^T & \cdots & x_0^T \\ x_d^T & x_{d-1}^T & \cdots & x_1^T \\ \vdots & \vdots & \ddots & \vdots \\ x_{T-2}^T & x_{T-3}^T & \cdots & x_{T-d-1}^T \\ x_{T-1}^T & x_{T-2}^T & \cdots & x_{T-d}^T \end{bmatrix}. \tag{4.21}$$

In this Section our objective is to establish the probability distribution of rows of X.

4.B.1 Single row of X

The autocovariance matrix of the original VAR process of order d in (4.20) is defined as $\Gamma(h) = \mathbb{E}[x_t x_{t+h}^T]$. Fourier transform of autocovariance matrix is called spectral density and is denoted as (for $i = \sqrt{-1}$)

$$\rho(\omega) = \sum_{h=-\infty}^{\infty} \Gamma(h) e^{-hi\omega}, \quad \omega \in [0, 2\pi]. \tag{4.22}$$

Inverse Fourier transform of the spectral density gives back the autocovariance matrix:

$$\Gamma(h) = \frac{1}{2\pi} \int_0^{2\pi} \rho(\omega) e^{hi\omega} d\omega, \quad h \in 0, \pm 1, \pm 2, \dots \tag{4.23}$$

For our VAR model in (4.20), the spectral density has a closed form expression [122]

$$\rho(\omega) = \left(I - \sum_{k=1}^{d} A_k e^{-ki\omega} \right)^{-1} \Sigma \left[\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega} \right)^{-1} \right]^* \in \mathbb{R}^{p \times p}, \tag{4.24}$$

where $*$ is the Hermitian of a matrix.

Let $X_{i,:}$ be any row vector of matrix X in (4.21), then

$$C_{\mathsf{X}} = \begin{bmatrix} \Gamma(0) & \Gamma(1) & \dots & \Gamma(d-1) \\ \Gamma(1)^T & \Gamma(0) & \dots & \Gamma(d-2) \\ \vdots & \vdots & \ddots & \vdots \\ \Gamma(d-1)^T & \Gamma(d-2)^T & \dots & \Gamma(0) \end{bmatrix} \in \mathbb{R}^{dp \times dp}. \tag{4.25}$$

Note that C_{X} is a block-Toeplitz matrix and so we can use the following property [121]

$$\inf_{\substack{1 \le j \le p \\ \omega \in [0, 2\pi]}} \Lambda_j[\rho(\omega)] \le \Lambda_k[C_V] \le \sup_{\substack{1 \le j \le p \\ \omega \in [0, 2\pi]}} \Lambda_j[\rho(\omega)], \quad \text{for } 1 \le k \le Kp. \tag{4.26}$$

Using (4.24), we can compute the lower bound. For this we use the following relationships: for any M, $||M||_2 = \sqrt{\Lambda_{\max}(M^T M)}$, and if M is symmetric, $||M||_2 = \Lambda_{\max}(M)$. Similarly, for any nonsingular M, $||M^{-1}||_2 = \frac{1}{\sqrt{\Lambda_{\min}(M^T M)}}$, and if M is symmetric,

$||M^{-1}||_2 = \frac{1}{\Lambda_{\min}(M)}$. Since $\rho(\omega)$ is symmetric, we have

$$\begin{aligned}\Lambda_{\max}[\rho(\omega)] &= \left|\left| \left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1} \Sigma \left[\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1}\right]^* \right|\right|_2 \\ &\leq \left|\left| \left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1} \right|\right|_2^2 ||\Sigma||_2 \\ &\leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]} \qquad (4.27)\end{aligned}$$

and the upper bound

$$\begin{aligned}\Lambda_{\min}[\rho(\omega)] &= \left[\left|\left| \left\{\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1} \Sigma \left[\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)^{-1}\right]^*\right\}^{-1} \right|\right|_2\right]^{-1} \\ &\geq \left[\left|\left| I - \sum_{k=1}^{d} A_k e^{-ki\omega} \right|\right|_2^2 ||\Sigma^{-1}||_2\right]^{-1} \\ &\geq \frac{\Lambda_{\min}(\Sigma)}{\Lambda_{\max}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]}. \qquad (4.28)\end{aligned}$$

Therefore, the C_{X} has the following bounds on its eigenvalues

$$\begin{aligned}&\frac{\Lambda_{\min}(\Sigma)}{\Lambda_{\max}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]} \leq \Lambda_k[C_{\mathsf{X}}] \leq \\ &\qquad\qquad \leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]},\end{aligned}$$

for $1 \leq k \leq dp,$ and $\omega \in [0, 2\pi]$.

Denoting $\Lambda_{\min}(\mathcal{A}) = \Lambda_{\min}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]$ for $\omega \in [0, 2\pi]$ and similarly $\Lambda_{\max}(\mathcal{A}) = \Lambda_{\max}\left[\left(I - \sum_{k=1}^{d} A_k^T e^{ki\omega}\right)\left(I - \sum_{k=1}^{d} A_k e^{-ki\omega}\right)\right]$ for $\omega \in [0, 2\pi]$, we can compactly write the above as

$$\frac{\Lambda_{\min}(\Sigma)}{\Lambda_{\max}(\mathcal{A})} \leq \Lambda_k[C_{\mathsf{X}}] \leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\mathcal{A})}, \qquad (4.29)$$

for $1 \leq k \leq dp$. From the above we extract the lower bound and denote it as

$$\Lambda_k[C_{\mathsf{X}}] \geq \frac{\Lambda_{\min}(\Sigma)}{\Lambda_{\max}(\mathcal{A})} = \mathcal{L}. \qquad (4.30)$$

4.B.2 All the rows of X

Consider a model obtained from the rows of matrix X (see (4.21)), i.e.,

$$\begin{bmatrix} x_{d-i+1} \\ x_{d-i} \\ \vdots \\ x_i \end{bmatrix} = \begin{bmatrix} A_1 & A_2 & \dots & A_{d-1} & A_d \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix} \begin{bmatrix} x_{d-i} \\ x_{d-i-1} \\ \vdots \\ x_{i-1} \end{bmatrix} + \begin{bmatrix} \epsilon_{d-i+1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Written in a compact form, the above expression takes the form

$$X_{j,:} = \mathbf{A} X_{j-1,:} + \mathcal{E}_j, \quad \text{for } j = 1, \dots, N,$$

which can be thought to be the transformations of the form

$$X_{1,:} = \begin{bmatrix} x_{d-1} \\ x_{d-2} \\ \vdots \\ x_0 \end{bmatrix} \quad \to \quad X_{2,:} = \begin{bmatrix} x_d \\ x_{d-1} \\ \vdots \\ x_1 \end{bmatrix} \quad \to \cdots \to \quad X_{N,:} = \begin{bmatrix} x_{N+d-2} \\ x_{N+d-3} \\ \vdots \\ x_{N-1} \end{bmatrix}.$$

Let

$$\mathcal{U} = \begin{bmatrix} X_{1,:} \\ \vdots \\ X_{N,:} \end{bmatrix} \in \mathbb{R}^{Ndp}, \tag{4.31}$$

be a vector composed from the output of the above VAR model during N steps. Then $C_{\mathcal{U}} \in \mathbb{R}^{Ndp \times Ndp}$ is the covariance matrix of vector $\mathcal{U}$

$$C_{\mathcal{U}} = \mathbb{E}(\mathcal{U}\mathcal{U}^T) = \mathbb{E} \begin{bmatrix} X_{1,:} \\ \vdots \\ X_{N,:} \end{bmatrix} \begin{bmatrix} X_{1,:}^T \dots X_{N,:}^T \end{bmatrix} = \begin{bmatrix} \mathbb{E}[X_{1,:}X_{1,:}^T] & \mathbb{E}[X_{1,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{1,:}X_{N,:}^T] \\ \mathbb{E}[X_{2,:}X_{1,:}^T] & \mathbb{E}[X_{2,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{2,:}X_{N,:}^T] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[X_{N,:}X_{1,:}^T] & \mathbb{E}[X_{N,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{N,:}X_{N,:}^T] \end{bmatrix}. \tag{4.32}$$

To establish the bounds on the eigenvalues of $C_{\mathcal{U}}$, we denote the spectral density of the corresponding VAR process as

$$\rho_{\mathsf{X}}(\omega) = \sum_{h=-\infty}^{\infty} \Gamma_{\mathsf{X}}(h) e^{-hi\omega}, \quad \omega \in [0, 2\pi],$$

where $\Gamma_{\mathsf{X}}(h) = \mathbb{E}[X_{j,:}X_{j+h,:}^T]$. Since $C_{\mathcal{U}}$ is a block-Toeplitz matrix, we can employ the same relationship as we used in Section 4.B.1

$$\inf_{\substack{1 \leq l \leq dp \\ \omega \in [0,2\pi]}} \Lambda_l[\rho_{\mathsf{X}}(\omega)] \leq \Lambda_k[C_{\mathcal{U}}] \leq \sup_{\substack{1 \leq l \leq dp \\ \omega \in [0,2\pi]}} \Lambda_l[\rho_{\mathsf{X}}(\omega)], \quad \text{for } 1 \leq k \leq Ndp. \tag{4.33}$$

In the following we establish the closed form expression of spectral density ρ_{X}. For this we write

$$\begin{aligned}
\rho_{\mathsf{X}}(\omega) &= \sum_{h=-\infty}^{\infty} \Gamma_{\mathsf{X}}(h) e^{-hi\omega} \\
&= \sum_{h=-\infty}^{\infty} \mathbb{E}[X_{j,:}X_{j+h,:}^T] e^{-hi\omega} \quad \text{for any } j \\
&= \sum_{h=-\infty}^{\infty} \mathbb{E}\left[\sum_{k=0}^{\infty} \mathbf{A}^k E_{j-k,:} \left(\sum_{s=0}^{\infty} \mathbf{A}^s E_{j+h-s,:}\right)^T\right] e^{-hi\omega} \\
&= \sum_{h=-\infty}^{\infty} \mathbb{E}\left[\sum_{k=0}^{\infty} \mathbf{A}^k E_{j-k,:} \left(\sum_{s=0}^{\infty} \mathbf{A}^{s-h} E_{j-s,:}\right)^T\right] e^{-hi\omega} \\
&= \sum_{h=-\infty}^{\infty} \sum_{k=0}^{\infty} \mathbf{A}^k \Sigma_E \left(\mathbf{A}^{k-h}\right)^T e^{-hi\omega} \\
&= \sum_{h=-\infty}^{\infty} \sum_{k=0}^{\infty} \mathbf{A}^k \Sigma_E \left(\mathbf{A}^{k-h}\right)^T e^{-hi\omega + ki\omega - ki\omega} \\
&= \sum_{h=-\infty}^{\infty} \sum_{k=0}^{\infty} \mathbf{A}^k e^{-ki\omega} \Sigma_E \left(\mathbf{A}^{k-h} e^{-(k-h)i\omega}\right)^* \\
&= \sum_{k=0}^{\infty} \mathbf{A}^k e^{-ki\omega} \Sigma_E \sum_{r=0}^{\infty} \left(\mathbf{A}^r e^{-ri\omega}\right)^* \\
&= \left(I - \mathbf{A}e^{-i\omega}\right)^{-1} \Sigma_E \left[\left(I - \mathbf{A}e^{-i\omega}\right)^{-1}\right]^*,
\end{aligned} \tag{4.34}$$

where we have used the fact that $\sum_{k=0}^{\infty} \mathbf{A}^k e^{-ki\omega} = \left(I - \mathbf{A}e^{-i\omega}\right)^{-1}$.

Now, using (4.33), (4.34), the results from Section 4.B.1 and the fact that the covariance matrix $\Sigma_{\mathcal{E}}$ has the form

$$\Sigma_{\mathcal{E}} = \begin{bmatrix} \Sigma & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix},$$

we can establish the following bounds

$$\frac{\Lambda_{\min}(\Sigma_{\mathcal{E}})}{\Lambda_{\max}\left[(I - \mathbf{A}^T e^{i\omega})\,(I - \mathbf{A}e^{-i\omega})\right]} \leq \Lambda_k[C_{\mathcal{U}}] \leq \frac{\Lambda_{\max}(\Sigma_{\mathcal{E}})}{\Lambda_{\min}\left[(I - \mathbf{A}^T e^{i\omega})\,(I - \mathbf{A}e^{-i\omega})\right]}.$$

Since $\Lambda_{\max}(\Sigma_{\mathcal{E}}) = \Lambda_{\max}(\Sigma)$, the upper bound becomes

$$\Lambda_{\max}[C_{\mathcal{U}}] \leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}\left[(I - \mathbf{A}^T e^{i\omega})\,(I - \mathbf{A}e^{-i\omega})\right]},$$

for $\omega \in [0, 2\pi]$. Denoting $\Lambda_{\min}(\boldsymbol{\mathcal{A}}) = \Lambda_{\min}\left[\left(I - \mathbf{A}^T e^{i\omega}\right)\left(I - \mathbf{A}e^{-i\omega}\right)\right]$ for $\omega \in [0, 2\pi]$, we can compactly write the above as

$$\Lambda_{\max}[C_{\mathcal{U}}] \leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})}. \tag{4.35}$$

4.B.3 Linear combination of rows of X

Consider a vector $q = Xa \in \mathbb{R}^N$ for any $a \in \mathbb{R}^{dp}$. Since each element $X_{i,:}^T a \sim \mathcal{N}(0, a^T C_{\mathsf{X}} a)$, it follows that $q \sim \mathcal{N}(0, Q_a)$ with a covariance matrix $Q_a \in \mathbb{R}^{N \times N}$, which is defined as

$$\begin{aligned}
Q_a = \mathbb{E}(qq^T) &= \mathbb{E}\begin{bmatrix} X_{1,:}^T a \\ \vdots \\ X_{N,:}^T a \end{bmatrix} \begin{bmatrix} a^T X_{1,:} \dots a^T X_{N,:} \end{bmatrix} \\
&= \begin{bmatrix} a^T\mathbb{E}[X_{1,:}X_{1,:}^T]a & a^T\mathbb{E}[X_{1,:}X_{2,:}^T]a & \dots & a^T\mathbb{E}[X_{1,:}X_{N,:}^T]a \\ a^T\mathbb{E}[X_{2,:}X_{1,:}^T]a & a^T\mathbb{E}[X_{2,:}X_{2,:}^T]a & \dots & a^T\mathbb{E}[X_{2,:}X_{N,:}^T]a \\ \vdots & \vdots & \ddots & \vdots \\ a^T\mathbb{E}[X_{N,:}X_{1,:}^T]a & a^T\mathbb{E}[X_{N,:}X_{2,:}^T]a & \dots & a^T\mathbb{E}[X_{N,:}X_{N,:}^T]a \end{bmatrix} \\
&= \begin{bmatrix} a^T & 0 & \dots & 0 \\ 0 & a^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a^T \end{bmatrix} \begin{bmatrix} \mathbb{E}[X_{1,:}X_{1,:}^T] & \mathbb{E}[X_{1,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{1,:}X_{N,:}^T] \\ \mathbb{E}[X_{2,:}X_{1,:}^T] & \mathbb{E}[X_{2,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{2,:}X_{N,:}^T] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[X_{N,:}X_{1,:}^T] & \mathbb{E}[X_{N,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{N,:}X_{N,:}^T] \end{bmatrix} \begin{bmatrix} a & 0 & \dots & 0 \\ 0 & a & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a \end{bmatrix} \\
&= (I_{N\times N} \otimes a^T) \begin{bmatrix} \mathbb{E}[X_{1,:}X_{1,:}^T] & \mathbb{E}[X_{1,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{1,:}X_{N,:}^T] \\ \mathbb{E}[X_{2,:}X_{1,:}^T] & \mathbb{E}[X_{2,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{2,:}X_{N,:}^T] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[X_{N,:}X_{1,:}^T] & \mathbb{E}[X_{N,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{N,:}X_{N,:}^T] \end{bmatrix} (I_{N\times N} \otimes a).
\end{aligned}$$

We denote the covariance matrix in the middle as

$$C_{\mathcal{U}} = \mathbb{E}(\mathcal{U}\mathcal{U}^T) = \mathbb{E}\begin{bmatrix} X_{1,:} \\ \vdots \\ X_{N,:} \end{bmatrix} \begin{bmatrix} X_{1,:}^T \dots X_{N,:}^T \end{bmatrix} = \begin{bmatrix} \mathbb{E}[X_{1,:}X_{1,:}^T] & \mathbb{E}[X_{1,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{1,:}X_{N,:}^T] \\ \mathbb{E}[X_{2,:}X_{1,:}^T] & \mathbb{E}[X_{2,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{2,:}X_{N,:}^T] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[X_{N,:}X_{1,:}^T] & \mathbb{E}[X_{N,:}X_{2,:}^T] & \dots & \mathbb{E}[X_{N,:}X_{N,:}^T] \end{bmatrix}. \tag{4.36}$$

Thus, we established that $q \sim \mathcal{N}(0, Q_a)$, where $Q_a = (I \otimes a^T)C_{\mathcal{U}}(I \otimes a)$.

In what follows, we compute trace(Q_a) and $||Q_a||_2$ for the covariance matrix Q_a. It can be seen that the trace of Q_a is given by

$$\text{trace}(Q_a) = Na^T C_{\mathsf{X}} a, \tag{4.37}$$

where C_{X} is defined in (4.6). Next, we compute upper bound on $||Q_a||_2$ as follows

$$\begin{aligned} ||Q_a||_2 &= ||(I \otimes a^T) C_{\mathcal{U}} (I \otimes a)||_2 \\ &\leq ||I \otimes a||_2^2 \; ||C_{\mathcal{U}}||_2 \\ &= ||a||_2^2 \; \Lambda_{\max}(C_{\mathcal{U}}), \end{aligned} \tag{4.38}$$

where the last equality follows since $||I \otimes a||_2^2 = \Lambda_{\max}\Big((I \otimes a^T)(I \otimes a)\Big) = \Lambda_{\max}\Big(I \otimes a^T a\Big) = ||a||_2^2$. We used a property of Kronecker product which states that for matrices with suitable dimensions, $(A \otimes B)(C \otimes D) = (AC \otimes BD)$.

To establish $\Lambda_{\max}(C_{\mathcal{U}})$, we use the results from Section 4.B.2, expression (4.35), which enable us to conclude that the upper bound of the largest eigenvalue of matrix $C_{\mathcal{U}}$ is given by

$$\Lambda_{\max}(C_{\mathcal{U}}) \leq \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})}.$$

Therefore, the bound on the covariance matrix $||Q_a||_2$ in (4.38) is now given by

$$||Q_a||_2 \leq ||a||_2^2 \; \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})} = \mathcal{M}. \tag{4.39}$$

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

4.C Bound on Regularization Parameter

To establish lower bound on the regularization parameter λ_N, we derive an upper bound on $R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}] \leq \alpha$, for some $\alpha > 0$, which will establish the required relationship $\lambda_N \geq \alpha \geq R^*[\frac{1}{N}Z^T\boldsymbol{\epsilon}]$. We will also utilize the notions of Gaussian width and covering net.

Definition 13 *For any set $\mathcal{S}$ and for a vector of independent zero-mean unit variance Gaussian variables $g \sim \mathcal{N}(0, I)$, the Gaussian width of the set is defined as*

$$w(\mathcal{S}) = \mathbb{E}_g[\sup_{u \in \mathcal{S}} \langle g, u \rangle]. \tag{4.40}$$

Denote $E_{:,j} \in \mathbb{R}^N$ as a column of matrix E and vector $U = [u_1^T, \dots, u_p^T]^T \in \mathbb{R}^{dp^2}$, where $u_i \in \mathbb{R}^{dp}$. Note that since $Z = I_{p \times p} \otimes X$, and $\boldsymbol{\epsilon} = \text{vec}(E)$, we can observe the following

$$\begin{aligned}
\sup_{R(U) \leq 1} \left\langle \frac{1}{N} Z^T \boldsymbol{\epsilon}, U \right\rangle &= \sup_{R(U) \leq 1} \frac{1}{N} \left\langle \left(I_{p \times p} \otimes X^T \right) \text{vec}(E), U \right\rangle \\
&= \sup_{R([u_1^T, \dots, u_p^T]^T) \leq 1} \frac{1}{N} \left(\left\langle X^T E_{:,1}, u_1 \right\rangle +, \dots, + \left\langle X^T E_{:,p}, u_p \right\rangle \right) \\
&= \frac{1}{N} \left(\sup_{R([u_1^T, \dots, u_p^T]^T) \leq 1} \left\langle X^T E_{:,1}, u_1 \right\rangle +, \dots, + \sup_{R([u_1^T, \dots, u_p^T]^T) \leq 1} \left\langle X^T E_{:,p}, u_p \right\rangle \right) \\
&= \frac{1}{N} \left(\sup_{R(u_1) \leq r_1} \left\langle X^T E_{:,1}, u_1 \right\rangle +, \dots, + \sup_{R(u_p) \leq r_p} \left\langle X^T E_{:,p}, u_p \right\rangle \right) \\
&= \frac{1}{N} \sum_{j=1}^{p} \sup_{R(u_j) \leq r_j} \left\langle X^T E_{:,j}, u_j \right\rangle
\end{aligned} \tag{4.41}$$

where $\sum_{j=1}^{p} r_j \leq 1$ and $r_j \geq 0$.

Our objective is to establish a high probability bound of the form

$$\mathbb{P}\left[\sup_{R(U) \leq 1} \left\langle \frac{1}{N} Z^T \boldsymbol{\epsilon}, U \right\rangle \leq \alpha \right] \geq \pi$$

where $0 \leq \pi \leq 1$, i.e., upper bound should hold with at least probability π. Using (4.41) and assuming that $\alpha = \sum_{j=1}^{p} \alpha_j$, we can rewrite the above probabilistic statement as follows

$$
\begin{aligned}
\mathbb{P}\left[\sup_{R(U)\leq 1}\left\langle\frac{1}{N}Z^T\boldsymbol{\epsilon}, U\right\rangle \leq \alpha\right] &= \mathbb{P}\left[\frac{1}{N}\sum_{j=1}^{p}\sup_{R(u_j)\leq r_j}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \sum_{j=1}^{p}\alpha_j\right] \\
&\geq \mathbb{P}\left[\left\{\sup_{R(u_1)\leq r_1}\frac{1}{N}\left\langle X^T E_{:,1}, u_1\right\rangle \leq \alpha_1\right\} \text{ and} \right. \qquad (4.42) \\
&\qquad \left. \ldots \text{ and } \left\{\sup_{R(u_p)\leq r_p}\frac{1}{N}\left\langle X^T E_{:,p}, u_p\right\rangle \leq \alpha_p\right\}\right] \\
&\geq \sum_{j=1}^{p}\mathbb{P}\left[\sup_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \alpha_j\right] - (p-1). \qquad (4.43)
\end{aligned}
$$

In the above derivations we used the observation that if $\left\{\sup\limits_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \alpha_j\right\}$, for each j hold, then the event $\left\{\sum_{j=1}^{p}\sup\limits_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \sum_{j=1}^{p}\alpha_j\right\}$ also holds but the reverse is not always true, implying that the probability space related to the event $\left\{\sum_{j=1}^{p}\sup\limits_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \sum_{j=1}^{p}\alpha_j\right\}$ is larger.

Therefore, based on (4.42), we see that we need to establish the following concentration bound

$$
\mathbb{P}\left[\sup_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle \leq \alpha_j\right] \geq \pi_j, \qquad (4.44)
$$

for each $j = 1, \ldots, p$.

In the following our objective would be to first establish that the random variable $\frac{1}{N}\left\langle X^T E_{:,j}, h\right\rangle$ has sub-exponential tails, where $h \in \mathbb{R}^{dp}$, $\|h\|_2 = 1$ is a unit norm vector. Based on the generic chaining argument we then use Theorem 1.2.7 in [113] and bound the expectation of the supremum of the original variable $\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$, i.e., bound $\mathbb{E}\left[\sup\limits_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right]$. Finally, using Theorem 1.2.9 in [113] we establish the high probability bound on how $\sup\limits_{R(u_j)\leq r_j}\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$ concentrates around its mean.

4.C.1 Martingale difference sequence

We start by writing

$$\big\langle X^T E_{:,j}, h\big\rangle = \langle E_{:,j}, Xh\rangle = \sum_{i=1}^{N} E_{i,j}, (X_{:,i}h) = \sum_{i=1}^{N} m_i,$$

where $m_i = E_{i,j}(X_{i,:}h)$, $i = 1, \ldots, N$. Observe that m_i is a martingale difference sequence (MDS), which can be shown by establishing that $\mathbb{E}(m_i|m_1, \ldots, m_{i-1}) = 0$ (see [79]). We can introduce a set $\{E_{1,:}, E_{2,:}, \ldots, E_{i-1,:}\} = \{\epsilon_d^T, \epsilon_{d+1}^T, \ldots, \epsilon_T^T\}$ and write

$$\mathbb{E}\big[m_i|m_1, \ldots, m_{i-1}\big] = \mathbb{E}\big[\mathbb{E}\big[m_i|m_1, \ldots, m_{i-1}, E_{1,:}, \ldots, E_{i-1,:}\big]\big],$$

using the technique of iterated expectation. Note that the set $\{E_{1,:}, E_{2,:}, \ldots, E_{i-1,:}\}$ contains more information than the set $\{m_1, \ldots, m_{i-1}\}$ and conditioning on it has fixed all the past history of the sequence until time stamp i. Since $m_i = E_{i,j}(X_{i,:}h)$, the terms $E_{i,j}$ and $X_{i,:}h$ are now independent. The independence follows since every row of matrix X is independent of the corresponding row of matrix E:

$$E = \begin{bmatrix} \epsilon_d^T \\ \epsilon_{d+1}^T \\ \vdots \\ \epsilon_{T-1}^T \\ \epsilon_T^T \end{bmatrix}, \quad X = \begin{bmatrix} x_{d-1}^T & x_{d-2}^T & \cdots & x_0^T \\ x_d^T & x_{d-1}^T & \cdots & x_1^T \\ \vdots & \vdots & \ddots & \vdots \\ x_{T-2}^T & x_{T-3}^T & \cdots & x_{T-d-1}^T \\ x_{T-1}^T & x_{T-2}^T & \cdots & x_{T-d}^T \end{bmatrix},$$

which can be verified by noting that the noise vector ϵ_{d+i} is independent from x_{d-k+i} since $(d+i) > (d-k+i)$ for $0 \le i \le T-d$ and $1 \le k \le d$. In other words, the information contained in x_{d-k+i} does not contain information from the noise ϵ_{d+i} (see (4.2)). Moreover,

$$\mathbb{E}\Big[m_i\Big] = \mathbb{E}\Big[E_{i,j}(X_{i,:}h)\Big] = \mathbb{E}\Big[E_{i,j}\Big]\mathbb{E}\Big[X_{i,:}h\Big] = 0, \tag{4.45}$$

due to the zero-mean noise $\mathbb{E}\big[E_{i,j}\big] = 0$. Consequently, we have shown that the conditional expectation $\mathbb{E}\big[m_i|m_1, \ldots, m_{i-1}, E_{1,:}, \ldots, E_{i-1,:}\big] = 0$ and therefore

$$\mathbb{E}\big[m_i|m_1, \ldots, m_{i-1}\big] = 0,$$

proving that $m_i = E_{i,j}(X_{i,:}h)$, $i = 1, \ldots, N$ is a martingale difference sequence.

Next, to show that $\frac{1}{N}\left\langle X^T E_{:,j}, h\right\rangle = \frac{1}{N}\sum_{i=1}^{N} m_i$ has sub-exponential tails, we first show that m_i is sub-exponential random variable and then use the proof argument similar to Azuma-type [126] and Bernstein-type [125] inequalities to establish that a sum over sub-exponential martingale difference sequence is itself sub-exponential.

4.C.2 Sub-exponential tails of $\frac{1}{N}\left\langle X^T E_{:,j}, h\right\rangle$

The MDS m_i is sub-exponential since it is a product of two Gaussians. Indeed, recall that E_{ij} and $X_{i,:}h$ are both Gaussian random variables, independent of each other. Employing a union bound enables us to write for any $\tau > 0$

$$\begin{aligned}
\mathbb{P}\Big[|m_i| \geq \tau\Big] &= \mathbb{P}\Big[|E_{ij}(X_{i,:}h)| \geq \tau\Big] \\
&\leq \mathbb{P}\Big[|E_{ij}| \geq \sqrt{\tau}\Big] + \mathbb{P}\Big[|X_{i,:}h| \geq \sqrt{\tau}\Big] \\
&\leq 2e^{-c_1\tau} + 2e^{-c_2\tau} \\
&\leq 4e^{-c\tau},
\end{aligned}$$

for some suitable constants $c_1 > 0$, $c_2 > 0$ and $c > 0$.

To establish that $\frac{1}{N}\sum_i m_i$ is sub-exponential, we note that the sub-exponential norm $\|\cdot\|_{\psi_1}$ (see [125], Definition 5.13) of m_i can be upper-bounded by a constant. We denote by $\kappa > 0$ the largest of these constants, i.e.,

$$\kappa = \max_{i=1,\ldots,N} \|m_i\|_{\psi_1} = \max_{i=1,\ldots,N} \|X_{i,:}h\|_{\psi_1}.$$

Now, using Lemma 5.15 in [125], the moment generating function of m_i satisfies the following result: for s such that $|s| \leq \frac{\eta}{\kappa}$ and for all $i = 1,\ldots,N$

$$\mathbb{E}\Big[e^{sm_i}\Big] \leq e^{cs^2\kappa^2}, \tag{4.46}$$

where c and η are absolute constants. Next, using Markov inequality, we can write for any $\varepsilon' > 0$

$$\begin{aligned}
\mathbb{P}\left[\sum_{i=1}^{N} m_i \geq \varepsilon'\right] &= \mathbb{P}\left[\exp\left(s\sum_{i=1}^{N} m_i\right) \geq \exp(s\varepsilon')\right] \\
&\leq \frac{\mathbb{E}\left[\exp\left(s\sum_{i=1}^{N} m_i\right)\right]}{\exp(s\varepsilon')}.
\end{aligned} \tag{4.47}$$

To bound the numerator, we use (4.46) and write for $|s| \leq \frac{\eta}{\kappa}$ utilizing the iterated expectation

$$\begin{aligned}
\mathbb{E}\left[\exp\left(s\sum_{i=1}^{N} m_i\right)\right] &= \mathbb{E}\left[\exp(sm_N)\exp\left(s\sum_{i=1}^{N-1} m_i\right)\right] \\
&= \mathbb{E}_{m_1,\ldots,m_{N-1}}\left[\mathbb{E}_{m_N|m_1,\ldots,m_{N-1}}\left[\exp(sm_N)\exp\left(s\sum_{i=1}^{N-1} m_i\right)\right]\right] \\
&= \mathbb{E}_{m_1,\ldots,m_{N-1}}\left[\mathbb{E}_{m_N|m_1,\ldots,m_{N-1}}\left[\exp(sm_N)\right]\exp\left(s\sum_{i=1}^{N-1} m_i\right)\right] \\
&\overset{\text{using (4.46)}}{\leq} \exp(cs^2\kappa^2)\mathbb{E}_{m_1,\ldots,m_{N-1}}\left[\exp\left(s\sum_{i=1}^{N-1} m_i\right)\right] \\
&\leq \exp(cs^2\kappa^2)\exp(cs^2\kappa^2)\mathbb{E}_{m_1,\ldots,m_{N-2}}\left[\exp\left(s\sum_{i=1}^{N-2} m_i\right)\right] \\
&\vdots \\
&\leq \exp(Ncs^2\kappa^2)
\end{aligned}$$

Substituting back to (4.47), we get for $|s| \leq \frac{\eta}{\kappa}$

$$\mathbb{P}\left[\sum_{i=1}^{N} m_i \geq \varepsilon'\right] \leq \exp(-s\varepsilon' + Ncs^2\kappa^2). \tag{4.48}$$

We now select s to minimize the right hand side of (4.48). For this, note that if the minimum is achieved for an s, which satisfies $|s| \leq \frac{\eta}{\kappa}$, then we simply minimize $-s\varepsilon' + Ncs^2\kappa^2$ and get $s = \frac{\varepsilon'}{N2c\kappa^2}$. On the other hand, if the minimum is achieved for an s outside the range $|s| \leq \frac{\eta}{\kappa}$, we pick the one on boundary $s = \frac{\eta}{\kappa}$. Thus, choosing $s = \min\left(\frac{\varepsilon'}{N2c\kappa^2}, \frac{\eta}{\kappa}\right)$, we obtain

$$\mathbb{P}\left[\sum_{i=1}^{N} m_i \geq \varepsilon'\right] \leq \exp\left(-\min\left(\frac{\varepsilon'^2}{4cN\kappa^2}, \frac{\eta\varepsilon'}{2\kappa}\right)\right).$$

Finally, setting $\varepsilon' = N\varepsilon$, for a suitable constant $c > 0$, we get

$$\mathbb{P}\left[\frac{1}{N}\sum_{i=1}^{N} m_i \geq \varepsilon\right] \leq \exp\left(-c\min\left(\frac{N\varepsilon^2}{\kappa^2}, \frac{N\varepsilon}{\kappa}\right)\right).$$

Repeating the above argument for $-\frac{1}{N}\sum_{i=1}^{N} m_i$, we obtain same bound and a combination of both of them gives the required concentration inequality for the sum over the martingale difference sequence

$$\mathbb{P}\left[\frac{1}{N}\left|\sum_{i=1}^{N} m_i\right| \geq \varepsilon\right] = \mathbb{P}\left[\frac{1}{N}\left|\left\langle X^T E_{:,j}, h\right\rangle\right| \geq \varepsilon\right] \leq 2\exp\left(-c\min\left(\frac{N\varepsilon^2}{\kappa^2}, \frac{N\varepsilon}{\kappa}\right)\right). \tag{4.49}$$

4.C.3 Establishing bound on the mean of supremum of $\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$

To establish a high probability bound on $\mathbb{E}\left[\sup\limits_{R(u_j)\leq r_j} \frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right]$, we use a generic chaining argument from [113], in particular Theorem 1.2.7 in [127]. For this, we define $(Y_{u_j})_{u_j\in R(u_j)\leq r_j} = \frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$ and $(Y_{v_j})_{v_j\in R(v_j)\leq r_j} = \frac{1}{N}\left\langle X^T E_{:,j}, v_j\right\rangle$ to be two centered random symmetric process, indexed by a fixed vectors u_j and v_j, respectively. They are centered due to (4.45) and they are symmetric since, for example, the process $(Y_{u_j})_{u_j\in R(u_j)\leq r_j}$ has the same law as process $\Big(-(Y_{u_j})_{u_j\in R(u_j)\leq r_j}\Big)$ (see the results established in (4.49)). Consider now the absolute difference of these two processes

$$\begin{aligned}\left|(Y_{u_j})_{u_j\in R(u_j)\leq r_j} - (Y_{v_j})_{v_j\in R(v_j)\leq r_j}\right| &= \frac{1}{N}\left|\left\langle X^T E_{:,j}, u_j - v_j\right\rangle\right| \\ &= \|u_j - v_j\|_2 \frac{1}{N}\left|\left\langle X^T E_{:,j}, \frac{u_j - v_j}{\|u_j - v_j\|_2}\right\rangle\right|.\end{aligned}$$

Using now the bound obtained in (4.49), we get

$$\begin{aligned}&\mathbb{P}\left[\frac{1}{N}\left|\left\langle X^T E_{:,j}, \frac{u_j - v_j}{\|u_j - v_j\|_2}\right\rangle\right| \geq \varepsilon\right] \\ =&\mathbb{P}\left[\|u_j - v_j\|_2\frac{1}{N}\left|\left\langle X^T E_{:,j}, \frac{u_j - v_j}{\|u_j - v_j\|_2}\right\rangle\right| \geq \|u_j - v_j\|_2\varepsilon\right] \\ =&\mathbb{P}\left[\frac{1}{N}\left|\left\langle X^T E_{:,j}, u_j - v_j\right\rangle\right| \geq \tau\right] \leq 2\exp\left(-c\min\left(\frac{N\tau^2}{\|u_j - v_j\|_2^2\kappa^2}, \frac{N\tau}{\|u_j - v_j\|_2\kappa}\right)\right),\end{aligned}$$

where $\tau = \|u_j - v_j\|_2\varepsilon$. Then, according to Theorem 1.2.7 in [127], we obtain the following bound on the expectation of the supremum of the difference between the processes

$$\mathbb{E}\left[\sup_{R(u_j)\leq r_j, R(v_j)\leq r_j} \frac{1}{N}\left|\left\langle X^T E_{:,j}, u_j\right\rangle - \left\langle X^T E_{:,j}, v_j\right\rangle\right|\right]$$
$$\leq c\left(\gamma_1\left(S_j, \frac{\|u_j - v_j\|_2}{N}\right) + \gamma_2\left(S_j, \frac{\|u_j - v_j\|_2}{\sqrt{N}}\right)\right), \qquad (4.50)$$

where c is a constant, $f_i(S_j, d_i)$, $i = 1, 2$, are the majorizing measures, which are defined in [113], Definition 1.2.5; $d_1 = \frac{\|u_j - v_j\|_2}{N}$ and $d_2 = \frac{\|u_j - v_j\|_2}{\sqrt{N}}$ are the distance measures on the set S_j defined for all vectors $s \in S_j : R(s) \leq r_j$. The definition of majorizing measure is as follows, for $\alpha > 0$

$$\gamma_\alpha(S_j, d) = \inf \sup_t \sum_{k\geq 0} 2^{\frac{k}{\alpha}} \Delta(A_k(t)), \qquad (4.51)$$

where inf is taken over all possible admissible sequences of the set S_j; $\Delta(A_k(t))$ denotes the diameter of element $A_k(t)$ with respect to the distance metric d defined as

$$\Delta(A_k(t)) = \sup_{t_1, t_2 \in A_k(t)} d(t_1, t_2), \qquad (4.52)$$

and $A_k(t) \in \mathcal{A}_k$ is an element of an admissible sequence in generic chaining, see Definition 1.2.3 in [113] for a detailed discussion on how $\mathcal{A}_k$ are constructed.

Observe that from definition of a diameter $\Delta(\cdot)$ in (4.52) and majorizing measure in (4.51) we can immediately see that for any constant $c > 0$

$$\gamma_\alpha\left(S_j, cd\right) = c\gamma_\alpha\left(S_j, d\right), \qquad (4.53)$$

since $\inf \sup\limits_t \sum_{k\geq 0} 2^{\frac{k}{\alpha}} \sup\limits_{t_1, t_2 \in A_k(t)} cd(t_1, t_2) = c \inf \sup\limits_t \sum_{k\geq 0} 2^{\frac{k}{\alpha}} \sup\limits_{t_1, t_2 \in A_k(t)} d(t_1, t_2)$. Moreover, in the next result we establish the following useful Lemma which would enable us to bound the γ_1 with the square of γ_2.

Lemma 14 *Given a metric space (S_j, d), we have*

$$\gamma_1(S_j, \|.\|_2) \leq \gamma_2^2(S_j, \|.\|_2). \qquad (4.54)$$

To prove this Lemma, we define $d(s,t) = \|s-t\|_2$. We use the traditional definition of majorizing measure $\gamma'_\alpha(S_j, d)$ from [128], equation (1.2):

$$\gamma'_\alpha(S_J, d) = \inf \sup_{s \in S} \left(\int_0^\infty \left(\log \frac{1}{\mu(B_d(s,\varepsilon))} \right)^{1/\alpha} d\varepsilon \right),$$

where $B_d(s,\varepsilon)$ is the closed ball of center t and radius ε based on the distance d and the infimum is taken over all the probability measure μ on S_j.

Note that $\gamma'_\alpha(S_j, d)$ relates to the majorizing measure $\gamma_\alpha(S_j, d)$ used in (4.50) as (see [128], Theorem 1.2)

$$K(\alpha)^{-1}\gamma_\alpha(S_j, d) \leq \gamma'_\alpha(S_j, d) \leq K(\alpha)\gamma_\alpha(S_j, d),$$

where $K(\alpha)$ is a constant depending on α only. As a result, it is enough to show that $\gamma'_1(S_j, d) \leq {\gamma'_2}^2(S_j, d)$. The required relationship is then established as follows

$$\begin{aligned}
\gamma'_1(S_j, d) &= \inf \sup_{t} \left(\int_0^\infty \left(\log \frac{1}{\mu(B_d(t,\varepsilon))} \right) d\varepsilon \right) \\
&\leq \inf \sup_{t} \left(\int_0^\infty \left(\log \frac{1}{\mu(B_d(t,\varepsilon))} \right)^{1/2} d\varepsilon \right)^2 \\
&= {\gamma'_2}^2(S_j, d).
\end{aligned}$$

And this completes the proof. Now using Theorem 2.1.1 in [113], and the definition of $\gamma_\alpha(S_j, d)$ in (4.51) we can establish that

$$\begin{aligned}
\gamma_2\left(S_j, \frac{\|.\|_2}{\sqrt{N}}\right) &= \frac{1}{\sqrt{N}}\gamma_2(S_j, \|.\|_2) \quad \text{using (4.53)} \\
&\leq \frac{1}{\sqrt{N}}\mathbb{E}\Big[\sup_{R(z)\leq r_j} \langle g, z\rangle\Big] \quad \text{using Theorem 2.1.1 in [113]} \\
&= r_j\frac{1}{\sqrt{N}}\mathbb{E}\Big[\sup_{R(u)\leq 1} \langle g, u\rangle\Big] \\
&= r_j\frac{1}{\sqrt{N}}w(\Omega_R), \qquad (4.55)
\end{aligned}$$

where the third equality follows since $\mathbb{E}\Big[\sup_{R(z)\leq r_j} \langle g, z\rangle\Big] = r_j\mathbb{E}\Big[\sup_{R(u)\leq 1} \langle g, u\rangle\Big]$ for $z = r_j u$, and in the last line we used the description of Gaussian width in Definition 13. Using

Lemma 14 and (4.53) above, we also get

$$\begin{aligned}\gamma_1\left(S_j, \frac{\|.\|_2}{N}\right) &= \frac{1}{N}\gamma_1\left(S_j, \|.\|_2\right) \quad \text{using (4.53)} \\ &\leq \frac{1}{N}\gamma_2^2\left(S, \|.\|_2\right) \quad \text{using Lemma 14} \\ &\leq r_j^2\frac{1}{N^2}w^2(\Omega_R) \quad \text{using (4.55)} \\ &\leq r_j\frac{1}{N^2}w^2(\Omega_R), \end{aligned} \tag{4.56}$$

where in the last line we used the fact that $r_j < 1$. Finally, substituting (4.55) and (4.56) into (4.50) and using Lemma 1.2.8 in [127], we get

$$\begin{aligned}\mathbb{E}\left[\sup_{R(u_j)\leq r_j, R(v_j)\leq r_j}\frac{1}{N}\left|\left\langle X^T E_{:,j}, u_j\right\rangle - \left\langle X^T E_{:,j}, v_j\right\rangle\right|\right] &= \mathbb{E}\left[\sup_{R(u_j)\leq r_j}\left|\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right|\right] \\ &\leq c r_j\left(\frac{w(\Omega_R)}{\sqrt{N}} + \frac{w^2(\Omega_R)}{N^2}\right).\end{aligned} \tag{4.57}$$

4.C.4 Establishing high probability concentration bound

Next, in order to establish the high probability concentration of the supremum of the random variable $\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle$ around its mean, we use Theorem 1.2.9 from [113]. For any $\epsilon_1 > 0$ and $\epsilon_2 > 0$, we have

$$\begin{aligned}\mathbb{P}\left[\sup_{R(u_j)\leq r_j}\left|\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right| \geq \mathbb{E}\left[\sup_{R(u_j)\leq r_j}\left|\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right|\right] + \epsilon_1 D_1 + \epsilon_2 D_2\right] \\ \leq c\exp(-\min(\epsilon_2^2, \epsilon_1)).\end{aligned} \tag{4.58}$$

where $D_i \leq \gamma_i(S_j, d)$, $i = 1, 2$, where $\gamma_i(S_j, d)$ are as defined in the discussion after (4.50). Therefore, using the result (4.57), the concentration inequality (4.58) can now be written as

$$\begin{aligned}\mathbb{P}\left[\sup_{R(u_j)\leq r_j}\left|\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right| \geq \left(c_2(1+\epsilon_2)r_j\frac{w(\Omega_R)}{\sqrt{N}} + c_1(1+\epsilon_1)r_j\frac{w^2(\Omega_R)}{N^2}\right)\right] \\ \leq c\exp(-\min(\epsilon_2^2, \epsilon_1)).\end{aligned} \tag{4.59}$$

To adapt to the form required in (4.44), we reverse the direction of inequality

$$\mathbb{P}\left[\sup_{R(u_j)\le r_j}\left|\frac{1}{N}\left\langle X^T E_{:,j}, u_j\right\rangle\right| \le \left(c_2(1+\epsilon_2)r_j\frac{w(\Omega_R)}{\sqrt{N}} + c_1(1+\epsilon_1)r_j\frac{w^2(\Omega_R)}{N^2}\right)\right]$$
$$\ge 1 - c\exp(-\min(\epsilon_2^2, \epsilon_1)). \quad (4.60)$$

4.C.5 Overall bound

Now we can combine the results obtained in (4.60) for each $j = 1, \ldots, p$ using the fact that $\sum_{j=1}^{p} r_j \le 1$ and using the form of the overall bound in (4.42). Therefore, we get

$$\mathbb{P}\left[\sup_{R(U)\le 1}\left\langle\frac{1}{N}Z^T\boldsymbol{\epsilon}, U\right\rangle \le \left(c_2(1+\epsilon_2)\frac{w(\Omega_R)}{\sqrt{N}} + c_1(1+\epsilon_1)\frac{w^2(\Omega_R)}{N^2}\right)\right]$$
$$\ge 1 - c\exp(-\min(\epsilon_2^2, \epsilon_1) + \log(p)).$$

This concludes our proof on establishing the bound on the regularization parameter.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

4.D Restricted Eigenvalue Condition

To establish restricted eigenvalue (RE) condition, we need to show that $\frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \ge \sqrt{\kappa N}$, $\kappa > 0$, for all $\Delta = \hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*$, $\Delta \in \text{cone}(\Omega_E)$, where cone$(\Omega_E)$ denotes a cone of an error set
$\Omega_E = \left\{\Delta \in \mathbb{R}^{dp^2}\middle| R(\boldsymbol{\beta}^* + \Delta) \le R(\boldsymbol{\beta}^*) + \frac{1}{c}R(\Delta)\right\}$. To show $\frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \ge \sqrt{\kappa N}$ for all $\Delta \in \text{cone}(\Omega_E)$, we will show that $\inf_{\Delta\in\text{cone}(\Omega_E)} \frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \ge \sqrt{\rho}$, for some $\rho > 0$ and then set $\kappa N = \rho$.

Note that the error vector can be written as $\Delta = [\Delta_1^T, \Delta_2^T, \ldots, \Delta_p^T]^T$, where Δ_i is of size $dp \times 1$. Also let $\boldsymbol{\beta}^* = [\beta_1^{*T}\beta_2^{*T}\ldots\beta_p^{*T}]^T$, for $\beta_i^* \in \mathbb{R}^{dp}$, then using our assumption in (4.4) that the norm $R(\cdot)$ is decomposable, we can represent original set Ω_E as a Cartesian product of subsets Ω_{E_i}, i.e., $\Omega_E = \Omega_{E_1} \times \Omega_{E_2} \times \cdots \times \Omega_{E_p}$, where

$$\Omega_{E_i} = \left\{ \Delta_i \in \mathbb{R}^{dp} \middle| R(\beta_i^* + \Delta_i) \leq R(\beta_i^*) + \frac{1}{c} R(\Delta_i) \right\},$$

which also implies that $\text{cone}(\Omega_E) = \text{cone}(\Omega_{E_1}) \times \text{cone}(\Omega_{E_2}) \times \cdots \times \text{cone}(\Omega_{E_p})$. Also, if $||\Delta||_2 = 1$, then we denote $||\Delta_i||_2 = \delta_i > 0$, so that $\sum_{i=1}^p \delta_i^2 = 1$. With this information, we can write

$$\begin{aligned}
\inf_{\Delta \in \text{cone}(\Omega_E)} \frac{||(I_{p\times p} \otimes X)\Delta||_2^2}{||\Delta||_2^2} &= \inf_{\substack{\Delta \in \text{cone}(\Omega_E) \\ ||\Delta||_2 = 1}} ||(I_{p\times p} \otimes X)\Delta||_2^2 \\
&= \inf_{\substack{\Delta \in \text{cone}(\Omega_E) \\ ||\Delta||_2 = 1}} ||X\Delta_1||_2^2 + ||X\Delta_2||_2^2 + \ldots + ||X\Delta_p||_2^2 \\
&= \sum_{i=1}^{p} \inf_{\substack{\Delta_i \in \text{cone}(\Omega_{e_i}) \\ ||\Delta_i||_2 = \delta_i}} ||X\Delta_i||_2^2. \qquad (4.61)
\end{aligned}$$

Our objective is to establish a high probability bound of the form

$$\mathbb{P}\left[\inf_{\Delta \in \text{cone}(\Omega_E)} \frac{||(I_{p\times p} \otimes X)\Delta||_2}{||\Delta||_2} \geq \rho \right] \geq \pi$$

where $0 \leq \pi \leq 1$, i.e., lower bound should hold with at least probability π. Note that if we square the terms inside the probability statement above, the probability of the resulting expression does not change since the squared terms are positive. Therefore, using (4.61) and assuming that $\rho^2 = \sum_{i=1}^{p} \rho_i^2$ we can rewrite the above as follows

$$\begin{aligned}
\mathbb{P}&\left[\inf_{\Delta \in \text{cone}(\Omega_E)} \frac{||(I_{p\times p} \otimes X)\Delta||_2}{||\Delta||_2} \geq \rho\right] \\
&= \mathbb{P}\left[\inf_{\Delta \in \text{cone}(\Omega_E)} \frac{||(I_{p\times p} \otimes X)\Delta||_2^2}{||\Delta||_2^2} \geq \sum_{i=1}^{p} \rho_i^2\right] \\
&= \mathbb{P}\left[\sum_{i=1}^{p} \inf_{\substack{\Delta_i \in \text{cone}(\Omega_{E_i}) \\ ||\Delta_i||_2 = \delta_i}} ||X\Delta_i||_2^2 \geq \sum_{i=1}^{p} \rho_i^2\right] \quad \text{using (4.61)} \\
&\geq \mathbb{P}\Bigg[\left\{\inf_{\substack{\Delta_1 \in \text{cone}(\Omega_{E_1}) \\ ||\Delta_1||_2 = \delta_1}} ||X\Delta_1||_2^2 \geq \rho_i^2\right\} \text{ and} \\
&\qquad\qquad \ldots \text{ and } \left\{\inf_{\substack{\Delta_p \in \text{cone}(\Omega_{E_p}) \\ ||\Delta_p||_2 = \delta_p}} ||X\Delta_p||_2^2 \geq \rho_i^2\right\}\Bigg] \\
&\geq \sum_{i=1}^{p} \mathbb{P}\left[\inf_{\substack{\Delta_i \in \text{cone}(\Omega_{E_i}) \\ ||\Delta_i||_2 = \delta_i}} ||X\Delta_i||_2^2 \geq \rho_i^2\right] - (p-1) \\
&= \sum_{i=1}^{p} \mathbb{P}\left[\inf_{\substack{\Delta_i \in \text{cone}(\Omega_{E_i}) \\ ||\Delta_i||_2 = \delta_i}} ||X\Delta_i||_2 \geq \rho_i\right] - (p-1)
\end{aligned}$$

Then, taking square root, we get

$$\begin{aligned}
\sum_{i=1}^{p} \mathbb{P}&\left[\inf_{\substack{\Delta_i \in \text{cone}(\Omega_{E_i}) \\ ||\Delta_i||_2 = \delta_i}} \frac{||X\Delta_i||_2}{||\Delta_i||_2} \geq \frac{\rho_i}{||\Delta_i||_2}\right] - (p-1) \\
&= \sum_{i=1}^{p} \mathbb{P}\left[\inf_{u_i \in \text{cone}(\Omega_{E_i}) \cap S^{dp-1}} ||Xu_i||_2 \geq \frac{\rho_i}{\delta_i}\right] - (p-1) \quad (4.62)
\end{aligned}$$

where we defined $u_i = \frac{\Delta_i}{||\Delta_i||_2}$ and S^{dp-1} is a unit sphere. Therefore, if we denote $\Theta_i = \text{cone}(\Omega_{E_i}) \cap S^{dp-1}$, we need to establish a lower bound of the form

$$\mathbb{P}\left[\inf_{u_i \in \Theta_i} ||Xu_i||_2 \geq \rho_i'\right] \geq \pi_i, \tag{4.63}$$

where $\rho_i' = \frac{\rho_i}{\delta_i}$. In the following derivations we set $\Theta = \text{cone}(\Omega_{E_i}) \cap S^{dp-1}$ and $u = u_i$ for all $i = 1, \ldots, p$ since the specific index i is irrelevant.

4.D.1 Bound on the infimum of $||Xu||_2$

Using results from Appendix 4.B we can establish that $Xu \in \mathbb{R}^N$ is a Gaussian random vector, i.e., $Xu \sim \mathcal{N}(0, Q_u)$, where covariance matrix $Q_u = (I_{N\times N} \otimes u^T)C_{\mathcal{U}}(I_{N\times N} \otimes u)$, $C_{\mathcal{U}}$ is defined in (4.36), and $u \in \Theta$ is a fixed vector.

To establish $\inf\limits_{u\in\Theta} ||Xu||_2$, we invoke a generic chaining argument from [113], specifically Theorem 2.1.5. For this we let $(Z_u)_{u\in\Theta} = ||Xu||_2 - \mathbb{E}(||Xu||_2)$ and $(Z_v)_{v\in\Theta} = ||Xv||_2 - \mathbb{E}(||Xv||_2)$ be two centered symmetric random processes. They are centered since, for example, $\mathbb{E}\Big[(Z_u)_{u\in\Theta}\Big] = \mathbb{E}(||Xu||_2) - \mathbb{E}(||Xu||_2) = 0$, and they are symmetric due to the later result shown in (4.65).

Sub-gaussianity of the process $Z_u - Z_v$.
We can show that the process difference

$$(Z_u)_{u\in\Theta} - (Z_v)_{v\in\Theta} = \|u-v\|_2\left(\left\|X\frac{u-v}{\|u-v\|_2}\right\|_2 - \mathbb{E}\left(\left\|X\frac{u-v}{\|u-v\|_2}\right\|_2\right)\right) \tag{4.64}$$

is a sub-Gaussian random process. This is indeed the case since we can establish that for $Z = ||X\frac{u-v}{\|u-v\|_2}||_2 - \mathbb{E}(||X\frac{u-v}{\|u-v\|_2}||_2)$, the sub-gaussian norm $\|Z\|_{\psi_2} \le K$ for some constant $K > 0$ (see [125], Definition 5.7). To show this, let $\xi = \frac{u-v}{\|u-v\|_2}$ and apply concentration of a Lipschitz function of Gaussian random variables. Specifically, observe that $X\xi \sim \mathcal{N}(0, Q_\xi)$ is distributed same as $\sqrt{Q_\xi}g \sim \mathcal{N}(0, Q_\xi)$, where $g \sim \mathcal{N}(0, I_{N\times N})$. Therefore, we can write

$$\mathbb{P}\Big[\,|\|X\xi\|_2 - \mathbb{E}(\|X\xi\|_2)| > \tau\Big] = \mathbb{P}\Big[\,\Big|\|\sqrt{Q_\xi}g\|_2 - \mathbb{E}(\|\sqrt{Q_\xi}g\|_2)\Big| > \tau\Big].$$

Moreover, note that $\|\sqrt{Q_\xi}g\|_2$ is a Lipschitz function with constant $\|\sqrt{Q_\xi}\|_2$ since we can write $\Big|\|\sqrt{Q_\xi}g_1\|_2 - \|\sqrt{Q_\xi}g_2\|_2\Big| \le \|\sqrt{Q_\xi}(g1 - g2)\|_2 \le \|\sqrt{Q_\xi}\|_2\ \|g_1 - g_2\|_2$. Using the concentration of a Lipschitz function of Gaussian random variables, we can obtain for all $\tau > 0$

$$\begin{aligned}\mathbb{P}\Big[\,|\|X\xi\|_2 - \mathbb{E}(\|X\xi\|_2)| > \tau\Big] &= \mathbb{P}\Big[\,\Big|\|\sqrt{Q_\xi}g\|_2 - \mathbb{E}(\|\sqrt{Q_\xi}g\|_2)\Big| > \tau\Big]\\ &\le 2\exp\left(-\frac{\tau^2}{2\|Q_\xi\|_2}\right)\\ &\le 2\exp\left(-\frac{\tau^2}{2\mathcal{M}}\right),\end{aligned} \tag{4.65}$$

where $||Q_\xi||_2 \leq ||\xi||_2^2 \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})} = \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})} = \mathcal{M}$ (see (4.39)), and which shows that $\|X\xi\|_2$ is sub-Gaussian with constant $K = \sqrt{\mathcal{M}}$.

Now, using (4.65) we can establish the sub-Gaussian tails of (4.64). Define $\tau' = \|u-v\|_2\tau$ and write

$$\begin{aligned}\mathbb{P}\Big[\Big|\|u-v\|_2\Big(\|X\xi\|_2 - \mathbb{E}(\|X\xi\|_2)\Big)\Big| > \|u-v\|_2\tau\Big] &= \mathbb{P}\Big[|(Z_u)_{u\in\Theta} - (Z_v)_{v\in\Theta}| > \tau'\Big] \\ &\leq 2\exp\left(-\frac{{\tau'}^2}{2\|u-v\|_2^2\mathcal{M}}\right). \qquad (4.66)\end{aligned}$$

Establishing bound on $\mathbb{E}\left(\inf\limits_{u\in\Theta}||Xu||_2\right)$ Using the results established in (4.66) and Theorem 2.1.5 in [113], we can conclude that the distance measure on the set Θ is $d(u,v) = \|u-v\|_2$ for $u, v \in \Theta$. Moreover, we can now obtain an upper bound on the expectation of the supremum of the process difference $|Z_u - Z_v|$

$$\begin{aligned}\mathbb{E}\left(\sup_{u,v\in\Theta}\Big|Z_u - Z_v\Big|\right) &= \mathbb{E}\left(\sup_{u,v\in\Theta}\Big|\ ||X(u-v)||_2 - \mathbb{E}(||X(u-v)||_2)\ \Big|\right) \\ &= \mathbb{E}\left(\sup_{u\in\Theta}\Big|\ ||Xu||_2 - \mathbb{E}(||Xu||_2)\ \Big|\right) \qquad \text{using Lemma 1.2.8 in [113]} \\ &\leq \mathbb{E}\Big[\sup_{u\in\Theta}\langle g, u\rangle\Big] \\ &\leq cw(\Theta), \qquad (4.67)\end{aligned}$$

where $g \sim \mathcal{N}(0, I)$, $w(\Theta)$ is the Gaussian width of set Θ and c is a constant.

Since we are interested in the bound on $\inf\limits_{u\in\Theta}||Xu||_2$, we can extract from (4.67) the lower bound on the expectation of the infimum of the process. Specifically, note that (4.67) can be written as

$$\mathbb{E}\left(\Big|\inf_{u\in\Theta}||Xu||_2 - \inf_{u\in\Theta}\mathbb{E}(||Xu||_2)\ \Big|\right) \leq \mathbb{E}\left(\sup_{u\in\Theta}\Big|\ ||Xu||_2 - \mathbb{E}(||Xu||_2)\ \Big|\right) \leq cw(\Theta),$$

leading to

$$-cw(\Theta) \leq \mathbb{E}\left(\inf_{u\in\Theta}||Xu||_2 - \inf_{u\in\Theta}\mathbb{E}(||Xu||_2)\right) \leq cw(\Theta).$$

The lower bound then takes the form

$$\mathbb{E}\left(\inf_{u\in\Theta}||Xu||_2\right) \geq \inf_{u\in\Theta}\ \mathbb{E}(||Xu||_2) - cw(\Theta) \qquad (4.68)$$

Note that the vector Xu is distributed as $Xu \sim \mathcal{N}(0, Q_u)$, which is the same as a vector $\sqrt{Q_u}g \sim \mathcal{N}(0, Q_u)$ for $g \sim \mathcal{N}(0, I)$. Therefore, using results of Lemma I.2 from [129], we can extract the following inequality

$$\left| \sqrt{\text{trace}(Q_u)} - \mathbb{E}(\|\sqrt{Q_u}g\|_2) \right| \leq 2\sqrt{\Lambda_{\max}(Q_u)}.$$

Moreover, based on our discussion, the same inequality holds for the random vector Xu since $\mathbb{E}(\|\sqrt{Q_u}g\|_2) = \mathbb{E}(\|Xu\|_2)$

$$\left| \sqrt{\text{trace}(Q_u)} - \mathbb{E}(\|Xu\|_2) \right| \leq 2\sqrt{\Lambda_{\max}(Q_u)}.$$

which leads to a lower bound on the expectation of the norm

$$\mathbb{E}(\|Xu\|_2) \geq \sqrt{\text{trace}(Q_u)} - 2\sqrt{\Lambda_{\max}(Q_u)}. \tag{4.69}$$

We will lower-bound the first term on the right hand side of (4.69) and upper bound the second one. In particular, using (4.37) we write $\text{trace}(Q_u) = Nu^T C_{\mathsf{X}} u$ for any $u \in \Theta$ and bound

$$\begin{aligned} \text{trace}(Q_u) = Nu^T C_{\mathsf{X}} u &= N||C_{\mathsf{X}}^{\frac{1}{2}} u||_2^2 \\ &\geq N \inf_{u \in \Theta} \; u^T C_{\mathsf{X}} u \\ &\geq N \inf_{u \in \mathbb{R}^{dp}} \; u^T C_{\mathsf{X}} u = N\Lambda_{\min}(C_{\mathsf{X}}) \\ &\geq N \frac{\Lambda_{\min}(\Sigma)}{\Lambda_{\max}(\mathcal{A})} = N\mathcal{L}. \end{aligned} \tag{4.70}$$

Moreover, using (4.39), we bound

$$||Q_u||_2 \leq ||u||_2^2 \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})} = \frac{\Lambda_{\max}(\Sigma)}{\Lambda_{\min}(\boldsymbol{\mathcal{A}})} = \mathcal{M}. \tag{4.71}$$

Therefore, substituting (4.71) and (4.70) into (4.69), we get

$$\mathbb{E}(\|Xu\|_2) \geq \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}}.$$

Since $\mathbb{E}(\|Xu\|_2)$ is bounded from below, we can write

$$\inf_{u \in \Theta} \mathbb{E}(\|Xu\|_2) \geq \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}}. \tag{4.72}$$

Finally, substituting (4.72) in (4.68) gives us

$$\mathbb{E}\left(\inf_{u\in\Theta}||Xu||_2\right) \geq \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta). \tag{4.73}$$

Establishing concentration inequality of $\inf_{u\in\Theta}||Xu||_2$.
Now from Lemma 2.1.3 in [113] and the results in [109] we extract the form of the high probability concentration inequality of $\inf_{u\in\Theta}\ ||Xu||_2$ around its mean, for $\tau > 0$

$$\mathbb{P}\left[\inf_{u\in\Theta}||Xu||_2 \leq \mathbb{E}\left(\inf_{u\in\Theta}||Xu||_2\right) - \tau\right] \leq c_1\exp(-c_2\tau^2).$$

In order to bring the above expression into the form of (4.63), we write

$$\mathbb{P}\left[\inf_{u\in\Theta}||Xu||_2 \geq \mathbb{E}\left(\inf_{u\in\Theta}||Xu||_2\right) - \tau\right] \geq 1 - c_1\exp(-c_2\tau^2).$$

Substituting the bound on the expectation from (4.73) gives us

$$\mathbb{P}\left[\inf_{u\in\Theta}||Xu||_2 \geq \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \tau\right] \leq c_1\exp(-c_2\tau^2). \tag{4.74}$$

4.D.2 Overall bound

Observe that in (4.74) we established a bound for each $u_i = \frac{\Delta_i}{||\Delta_i||_2}$ of the form

$$\mathbb{P}\left[\inf_{\substack{\Delta_i\in\text{cone}(\Omega_{E_i})\\ ||\Delta_i||_2=\delta_i}}\frac{||X\Delta_i||_2}{||\Delta_i||_2} \geq ||\Delta_i||_2\ \rho'_i\right] \geq 1 - c_1\exp(-c_2\eta_i^2),$$

where $\rho'_i = \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta_i$. Then using the fact that $\rho_i = \rho'_i\delta_i,\ \rho^2 = \sum_{i=1}^p \rho_i^2$, $\sum_i^p \delta_i^2 = 1$ and setting $\eta_i = \eta$ for all $i = 1, \ldots, p$, we get

$$\rho^2 = \left[\sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta\right]^2 \sum_{i=1}^p \delta_i^2 = \left[\sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta\right]^2.$$

Taking the square root of the above and using (4.62) we finally get

$$\mathbb{P}\left[\inf_{\Delta\in\text{cone}(\Omega_E)}\frac{||(I_{p\times p}\otimes X)\Delta||_2}{||\Delta||_2} \geq \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \eta\right] \geq 1 - pc_1\exp(-c_2\eta^2). \tag{4.75}$$

Establishing bound on N.

Now setting $\eta = \varepsilon\sqrt{N\mathcal{L}}$ for $0 < \varepsilon < 1$, the right hand side of the inequality inside the probability statement in (4.75) must be equal to

$$\sqrt{\kappa N} = \sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta) - \varepsilon\sqrt{N\mathcal{L}} = \varepsilon'\sqrt{N\mathcal{L}} - 2\sqrt{\mathcal{M}} - cw(\Theta),$$

for some positive constant ε'. Since $\kappa N > 0$, it follows that we require

$$\varepsilon'\sqrt{N\mathcal{L}} > 2\sqrt{\mathcal{M}} + cw(\Theta),$$

or equivalently

$$\sqrt{N} > \frac{2\sqrt{\mathcal{M}} + cw(\Theta)}{\varepsilon'\sqrt{\mathcal{L}}} = \mathcal{O}(w(\Theta)).$$

This concludes our proof on establishing the restricted eigenvalue conditions. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Chapter 5

Semi-Markov Switching Vector Autoregressive Model: Heterogeneous Data Modeling

In this chapter we consider the problem of anomaly detection in full flight data, consisting of continuous and discrete parameters (see Figure 1.2). The goal is, similarly as before, to detect anomalous flight segments, due to mechanical, environmental, or human factors in order to identifying operationally significant events and provide insights into the flight operations and highlight otherwise unavailable potential safety risks and precursors to accidents. For this purpose, we propose a framework which represents each flight using a semi-Markov switching vector autoregressive (SMS-VAR) model, based on the combination of the ideas for modeling discrete data (HSMM) and continuous data (VAR). Detection of anomalies is then based on measuring dissimilarities between the model's prediction and data observation.

For this purpose, in Section 5.2 we present a detailed model description and parameter learning algorithm based on EM. In Section 5.3 we present the anomaly detection framework, which is based on the dissimilarity of one-step ahead predicted and filtered phase distributions, showing advantage over the standard likelihood-based approach. Finally, in Section 5.5 we provide extensive evaluations on synthetic and real data, including 20000 unlabeled flights, showing SMS-VAR outperforming many base line algorithms and accurately detecting different types of anomalies.

5.1 Introduction

To model heterogeneous flight data, consisting of a mixture of discrete and continuous parameters, we propose to merge the ideas of HSMM and VAR and use switching vector autoregressive model [130]. These are models which alternatively known as hybrid models, state-space models with switching, jump-linear systems, etc. [131], [132]. In these type of models, the system evolves according to a certain dynamics until it switches to another dynamics. The switching is usually discrete and is defined in terms of a (semi-) Markovian process.

The problem of anomaly detection based on heterogeneous data was addressed by several researchers. For example, the work of [133] presented a model-based framework to identify flight human-automation issues using switches data and sensor measurements. The anomaly is identified if there is a difference between inferred intents of the automation and the observed pilot actions. The limitation of this approach is that it assumes the data is noise-free and does not account for parameter uncertainties. Moreover, it is not clear how the algorithm performs on large flight data, since the presented evaluations are limited to a few examples.

The work of Li et. al. [134] proposed to detect anomalies in the flight data based on continuous and discrete features using a clustering approach, called ClusterAD. Their idea is to represent each flight as a vector, by concatenating all feature across time. After dimensionality reduction, the data is clustered based on Euclidean distance measure to identify outliers and groups of similar flights. A potential issue with this approach is a misalignment between time series from different flights. Since the size of the vectors needs to be the same for all flights, forcing such equality can introduce spurious dissimilarities, increasing false positives in the detected anomalies. Moreover, the study in [135] revealed that ClusterAD does not perform well on discrete anomalies as compared to MKAD. Partially, this is due to the use of Euclidean similarity measure on heterogeneous data vectors, making it less effective in finding discrete anomalies.

Another approach for heterogeneous anomaly detection in aviation systems based on multiple kernel learning (MKAD) was proposed by Das et. al. [74]. The method constructs a kernel matrix as a convex combination of a kernel over discrete sequences and continuous time series. One-class SVM [76] is then used to detect anomalies. Although the method usually shows good performance results, it lacks scalability since the kernel matrix has to be updated for each new flight.

Summarizing the above literature and comparing to the proposed switching vector autoregressive model, we can make several remarks. First, our method is unsupervised, thus it does not require a training set of labeled normal flights. Moreover, our framework works with data, where each data sample can be of variable length. As compared to data-driven methods, our framework is model-based and therefore can be computationally more efficient in the detection stage by not requiring to recompute the model for each test flight. At the same time, the model construction requires only basic knowledge about the considered parameters and can easily be extended to other anomaly detection domains.

To motivate our proposed approach, consider Figure 5.1, showing a real flight data and a model to represent it. On the left we plot a part of flight related to landing and show time series of several *pilot switches* (from top: thrust, altitude, autopilot, flight director, localizer) as well as some of the *sensors* (altitude, pitch angle, airspeed, longitudinal acceleration, fuel flow). The switches act as controls, determining the behavior of aircraft and sensors measure the effects of the controls on the system. A combination of the switches set by a pilot determines the aircraft's behavior for a certain period of time after which a different combination of switches defines another flight period and so on. Within each flight period, called a *phase* (shown in red on the left in Figure 5.1), the aircraft's dynamics usually remains consistent and steady, while across phases the dynamics change.

As an example, consider a lower right plot in Figure 5.1, showing aircraft's path as it descends from 10000 feet to a runway. The descent is interrupted by some event, causing it to fly back to a certain altitude, make a circle and repeat the landing. The path is partitioned into phases shown with different colors and numbers. For instance, phase 5 corresponds to a steady descent, where aircraft constantly loses altitude while maintaining its airspeed. This phases is interrupted by phase 2 of duration about 50 seconds, caused by a switch off in auto thrust system and activating hold altitude switch.

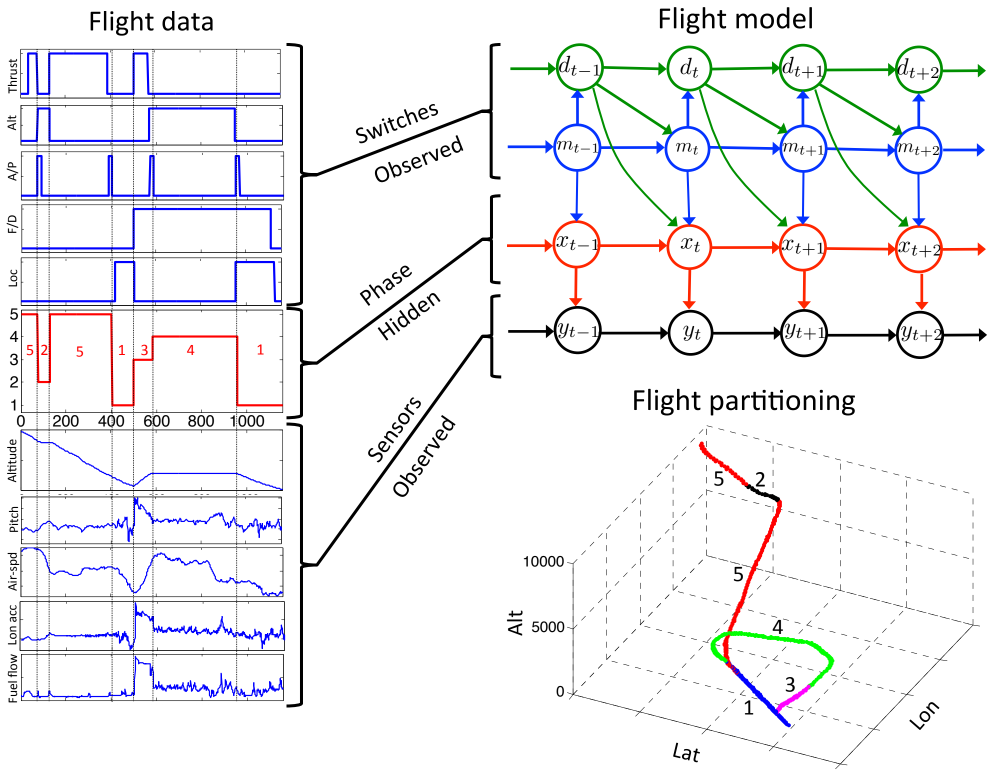


Figure 5.1: Representing heterogeneous flight data using SMS-VAR model. Left plot shows time evolution of several pilot switches, phase and sensor measurements during aircraft's landing. Top right plot shows the proposed model to represent such data. The bottom right graph shows the trajectory of aircraft, where its path is partitioned into phases. The value of the phases is obtained after constructing SMS-VAR on such data and running Viterbi algorithm to recover most probable phase path.

In phase 2 the aircraft levels off by steadily increasing its pitch angle and losing airspeed.

Thus, if a flight is partitioned into multiple phases, determined by the pilot controls (switches), then the continuous dynamics of each phase can be represented separately by its own model. We propose to model such data with a dynamic Bayesian network - *semi-Markov switching vector autoregressive* (SMS-VAR) model, shown on the right plot of Figure 5.1. We note that our motivation comes from a rich literature of systems identification [50], where a standard approach for modeling continuous system dynamics (in our case the flight's sensor measurements) is a vector autoregressive model (VAR) [79]. However, as we discussed above, using a single VAR model for the entire flight is inappropriate, thus we employ multiple VARs. A change from one VAR process to another, i.e., the switching behavior, is modeled with a hidden variable x_t, representing a flight phase.

To model the dynamics of flight switches, we convert their representation from binary categorical into discrete, i.e., at each time stamp t a vector of zeros and ones (values of switches at t) is converted into an integer. We call the resulting variable a flight mode m_t and represent it using semi-Markov model (SMM) [136]. SMM is an extension of a simple Markov chain, allowing to model arbitrary state durations. A simple Markov chain has implicit geometric state duration distribution [137], causing fast state transitions and is inappropriate for our case. SMM fixes this by introducing a variable d_t which controls the duration of mode m_t (see Section 5.2.1 for more details).

5.2 Semi-Markov Switching VAR

In this Section we formally introduce the SMS-VAR model and show details about the parameter learning algorithm.

5.2.1 Model Specification

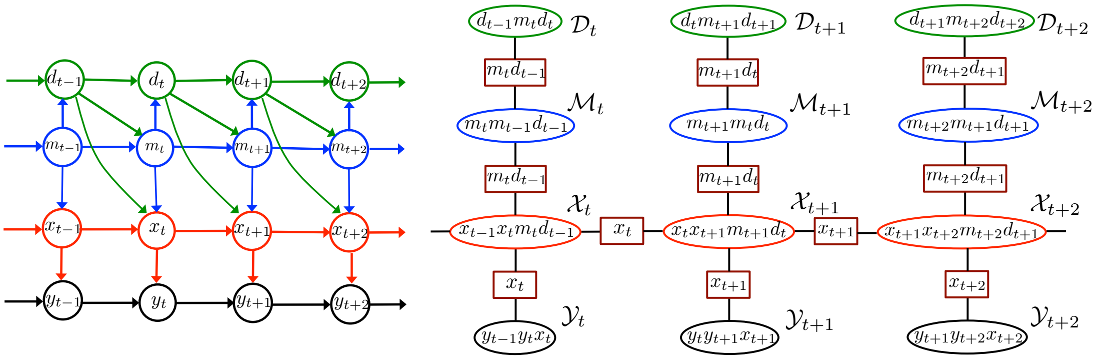


Figure 5.1: Left: Dynamic Bayesian Network of Semi-Markov Switching Vector Autoregressive Model (SMS-VAR). Right: Junction Tree for SMS-VAR. Ovals represent graph cliques, denoted by calligraphic letters $\mathcal{D}_t, \mathcal{M}_t, \mathcal{X}_t$, and $\mathcal{Y}_t$, rectangles denote separators. Symbols within shapes show variables on which the corresponding objects depend.

In this work we propose to model the flight of an aircraft using semi-Markov switching vector autoregressive model (SMS-VAR), whose Dynamic Bayesian Network (DBN) is shown in Figure 5.1. From the graphical model perspective, SMS-VAR has four classes of variables: continuous sensor measurements $y_t \in \mathbb{R}^{n_y}$, discrete phase $x_t \in \{1, \ldots, n_x\}$, discrete mode, $m_t \in \{1, \ldots, n_m\}$ and a positive real variable $d_t \in \mathbb{Z}_{\geq 0}$, determining the time duration of the mode and phase in a particular state. The SMS-VAR model is then fully defined by specifying the probability distributions which govern the time evolution of the above four variables. In particular, the probability distribution of mode transition is modeled as

$$p(m_t|m_{t-1}, d_{t-1}) = \begin{cases} p(m_t|m_{t-1}) & \text{if } d_{t-1} = 1 \\ \delta(m_t, m_{t-1}) & \text{if } d_{t-1} > 1 \end{cases}, \tag{5.1}$$

where $\delta(a, b)$ denotes the Dirac delta function: $\delta(a, b) = 1$ if $a = b$ and 0 otherwise. In essence, in this expression d_{t-1} works as a down counter for mode persistence, i.e., when $d_{t-1} > 1$, the mode is forced to remain unchanged: $m_t = m_{t-1}$. On the other hand, when $d_{t-1} = 1$, a new mode state m_t is determined by sampling from $p(m_t|m_{t-1}) \in \mathbb{R}^{n_m \times n_m}$, which is a 2-dimensional multinomial distribution. Note that to disallow self-transitions and ensure that mode changes to another state whenever $d_{t-1} = 1$, we set all the diagonal entries in $p(m_t|m_{t-1})$ to zero.

The duration variable d_t is modeled as

$$p(d_t|m_t, d_{t-1}) = \begin{cases} p(d_t|m_t) & \text{if } d_{t-1} = 1 \\ \delta(d_t, d_{t-1} - 1) & \text{if } d_{t-1} > 1 \end{cases}, \tag{5.2}$$

which means that as long as $d_{t-1} > 1$, we simply set $d_t = d_{t-1} - 1$. On the other hand, for $d_{t-1} = 1$, the new duration d_t is determined by sampling from $p(d_t|m_t)$, based on the current value of mode m_t. In this work we assume that

$$p(d_t|m_t) := P(d_t = k|m_t) = \frac{\lambda_{m_t}^k e^{-\lambda_{m_t}}}{k!}$$

is a Poisson distribution with $\lambda_{m_t} > 0$ and $k = 0, 1, 2, \ldots$. Observe that d_t together with m_t define a semi-Markov process [136] (top two chains shown in the left plot of Figure 5.1), which has more flexibility in modeling mode durations as opposed to a simple Markov process, whose implicit mode duration is constrained to a geometric distribution [137].

The phase x_t is distributed according to the following transition model

$$p(x_t|x_{t-1}, m_t, d_{t-1}) = \begin{cases} p(x_t|x_{t-1}, m_t) & \text{if } d_{t-1} = 1 \\ \delta(x_t, x_{t-1}) & \text{if } d_{t-1} > 1 \end{cases}. \tag{5.3}$$

Here $p(x_t|x_{t-1}, m_t) \in \mathbb{R}^{n_x \times n_x \times n_m}$ is a 3-dimensional multinomial distribution of x_t: for each value of the mode m_t, there is a separate transition matrix defining the distribution of x_t, given x_{t-1}: $p(x_t|x_{t-1})$. It is invoked whenever the counter $d_{t-1} = 1$, otherwise the phase is forced to stay in the same state. We note that in contrast to the mode distribution in (5.1), here we allow self-transitions even when $d_{t-1} = 1$, i.e., $p(x_t|x_{t-1}, m_t)$ for each m_t can have non-zero diagonal. The idea behind this modeling step comes from the motivation to enable long-lasting phase persistence. In other words, although the mode can switch quickly to another state, the phase, on the other hand, has a flexibility to either remain unchanged or transition to another phase. In fact, this property of our model is precisely what enables to compress many mode states into a single phase, allowing the use of only a few VAR processes to model the data.

Finally, the sensor measurements are modeled using first-order VAR process:

$$y_t = A_{x_t} y_{t-1} + \epsilon_{x_t}, \tag{5.4}$$

where $A_{x_t} \in \mathbb{R}^{n_y \times n_y}$ is the VAR transition matrix and $\epsilon_{x_t} \sim \mathcal{N}(0, \Sigma_{x_t})$ is a Gaussian noise, uncorrelated in time t. The probability distribution of y_t then takes the form

$$p(y_t|y_{t-1}, x_t) \propto Ce^{-\frac{1}{2}(y_t - A_{x_t} y_{t-1})^T \Sigma_{x_t}^{-1} (y_t - A_{x_t} y_{t-1})}, \tag{5.5}$$

for some normalization constant C. Note that for each value of the phase variable x_t, there is a separate VAR process with its own transition matrix and noise characteristics, which determines the time evolution of the vector y. In this work we assume that Σ_{x_t} is identity, thus the only unknown parameter in (5.5) is a transition matrix A_{x_t}. Each VAR process is assumed to be stable [79], i.e., all the eigenvalues of A_{x_t} have magnitude less than 1.

We remark on several important points about our model. (i) Note that SMS-VAR is similar but different from Markov switching autoregressive model [138], also known in literature as hybrid, switching state-space models or jump-linear systems [139]. The main difference is that the switching dynamics is governed by a hierarchy of observed, m_t, and unobserved, x_t semi-Markov processes, rather than a single unobserved Markov process. (ii) We could have defined our model without a phase x_t, where the mode m_t would directly determine the active VAR process. However, whenever any of the switches change its state, there would be a transition to a different VAR. This is a bad design since the transitions would be too frequent and the number of VARs would be too large. Including phase x_t, which depends on mode m_t and duration d_t, enables data compression since not every switch change would result in the phase change. This behavior is observed on the left of Figure 5.1, where phase 5 is insensitive to a change in auto thrust at $t = 50$, similarly phase 1 at $t = 1200$ stays same, although some switches change. (iii) Preliminary evaluations of the flight dataset based on correlogram [79] revealed that sample autocorrelation functions of the time series exhibit a fast decay beyond the first lag. i.e., there are no long-range interaction between two events far away from each other. Therefore, the use of first-order dependency in our model is adequate to represent the data. Based on this, we position SMS-VAR as a short-memory model [140] and the proposed anomaly detector in Section 5.3 specifically targets short-term anomaly events.

5.2.2 Parameter Learning

Given data $D = \{F^1, \dots, F^N\}$, consisting of N multivariate time series in the form $F^i = \{\bar{d}^i_1, \dots, \bar{d}^i_{T_i}, \bar{m}^i_1, \dots, \bar{m}^i_{T_i}, \bar{y}^i_1, \dots, \bar{y}^i_{T_i}\}$, (bar over the variable means that it is observed), our objective here is to estimate the parameters of SMS-VAR model:

$$\Theta = \{p(m_t|m_{t-1}), p(x_t|x_{t-1}, m_t), \lambda_{m_t}, A_{x_t}\}. \tag{5.6}$$

Since the data related to hidden phase $X^i = \{x^i_1, \dots, x^i_{T_i}\}$ is unobservable, the standard approach is to use Expectation-Maximization (EM) algorithm [18]. The idea is to find Θ, which maximizes likelihood of all the observed and unobserved data $p\Big(F^{1:N}, X^{1:N}\Big|\Theta\Big)$. Assuming that we have an initial estimate of parameters Θ_0, the EM algorithm consists of iterating the following two steps until convergence:

- *E*-step: $Q(\Theta,\Theta_k) = \mathbb{E}_{X^{1:N}}\Big[\log p\Big(F^{1:N}, X^{1:N}\Big|\Theta\Big)\Big|F^{1:N}, \Theta_k\Big]$

- M-step: $\Theta_{k+1} = \arg\max_{\Theta} Q(\Theta, \Theta_k)$.

Note that the E-step is executed for each flight independently, while in M-step the resulting probability information from all flights is collected to update the model parameters.

Also, observe that the execution of EM for the considered model can be challenging since the nodes in DBN are of mixed data type, complicating the inference in E and optimization in M steps. However, exploiting the fact that d_t, m_t and y_t are observable, we can compute both steps very efficiently.

***E*-step** Given the parameter specifications in Section 5.2.1, the E-step can be efficiently computed using Junction Tree algorithm [7]. Specifically, based on the DBN structure of the model on the left plot of Figure 5.1, we construct its junction tree (JT), shown on the right plot of Figure 5.1. JT is simply a tree-structured representation of the graph, which helps to decompose the global computations of joint probability $p\Big(F^{1:N}, X^{1:N}\Big|\Theta\Big)$ into a linked set of local computations.

Each oval node in junction tree in Figure 5.1, representing cliques in the graph of SMS-VAR, is initialized with a value of the corresponding probability distribution. For example, a node $\mathcal{D}_t$ is initialized with the value of duration distribution $p(\bar{m}_t|\bar{m}_{t-1}, \bar{d}_{t-1})$ in (5.2). Similarly, $\mathcal{M}_t$, $\mathcal{X}_t$, and $\mathcal{Y}_t$ are initialized by evaluating (5.1), (5.3) and (5.5) on the data.

After all the N trees are initialized with the data $D = \{F^1, \ldots, F^N\}$, the operation of Junction Tree algorithm to compute E-step consists of propagating messages forward in time (form $t = 1$ to $t = T_i$) and backward in time (form $t = T_i$ to $t = 1$). For example, at iteration k, the result of forward propagation is the computation of likelihood of data $p(F^{1:N}|\Theta_k)$, while the backward propagation computes $p(x_t, x_{t+1}|F^{1:N}, \Theta_k)$ and $p(x_t|F^{1:N}, \Theta_k)$ for each t.

The important practical aspect of the above calculations is to prevent numerical underflow, occurring during message propagation when many small numbers multiplied together. To avoid this, we performed all operations in log-scale and at each time stamp t normalized the messages to have their probability mass sum to one (using log-sum-exp function).

***M*-step** The result of E-step is now used to update the parameter estimates Θ in (5.6). We note that since phase x_t is the only unobservable part of the model, the only parameters that are re-estimated are phase transition distribution $p(x_t|x_{t-1}, m_t)$ and VAR transition matrices A_{x_t}. The other parameters, i.e., mode and duration distributions, depend on variables which are completely observable and estimated directly from data once and never re-estimated during EM iterations.

To estimate phase transition distribution $p(x_t|x_{t-1}, m_t)$, it can be shown that optimization problem $\arg\max_{\Theta} Q(\Theta, \Theta_k)$ amounts to estimating for each value of m_t the transition matrix $p(x_t|x_{t-1})$ by multiplying matrices $p(x_t, x_{t+1}|F^1\colon F^N, \Theta_k)$ across those time steps t at which m_t matches the mode value $\bar{m}_t$ in the data.

The solution of $\arg\max_{\Theta} Q(\Theta, \Theta_k)$ for A_{x_t} can be shown to be equivalent to a solution of a least-square problem for each $x_t \in \{1, \ldots, n_x\}$. Specifically, using $p(x_t|F^{1:N}, \Theta)$ from the results of E-step, we weight each sample vector $\bar{y}_t$ with a scalar $w_t = p(x_t|F^{1:N}, \Theta)$ for one of the x_t: $\bar{y}'_t = w_t\bar{y}_t$. Then for each weighted data sequence $\bar{y}'_t, \ldots, \bar{y}'_{T_i}$, $i = 1 \ldots, N$ we can stack the vectors as in expression (5.4) in a matrix form and write the following system of equations

$$\begin{bmatrix} \bar{y}'^T_2 & \bar{y}'^T_3 & \ldots & \bar{y}'^T_{T_i} \end{bmatrix}^T = \begin{bmatrix} \bar{y}'^T_1 & \bar{y}'^T_2 & \ldots & \bar{y}'^T_{T_i-1} \end{bmatrix}^T A^T_{x_t}.$$

In compact notations above can be written as $Y_i = M_i B$, where $Y_i, M_i \in \mathbb{R}^{L_i \times n_y}$ and $B = A^T_{x_t} \in \mathbb{R}^{n_y \times n_y}$ for $L_i = T_i - 1$. Now stacking together the equations for all N data sequences, we get $[Y^T_1, \ldots, Y^T_N]^T = [M^T_1, \ldots, M^T_N]^T B$, which again can be compactly written as a matrix equation $Y = MB$, where $Y, M \in \mathbb{R}^{L \times n_y}$ for $L = \sum_{i=1}^N L_i$. Now vectorizing (column-wise) each matrix in $Y = MB$, we get

$$\begin{aligned} \text{vec}(Y) &= (I_{n_y \times n_y} \otimes M)\text{vec}(Y) \\ \mathbf{y} &= Z\boldsymbol{\beta}, \end{aligned}$$

where now $\mathbf{y} \in \mathbb{R}^{Ln_y}$, $Z = (I_{n_y \times n_y} \otimes M) \in \mathbb{R}^{Ln_y \times n_y^2}$, $\boldsymbol{\beta} \in \mathbb{R}^{n_y^2}$ and $\otimes$ is the Kronecker product. Consequently, $\boldsymbol{\beta}$ (rows of A_{x_t} stacked in a vector) is estimated by solving

$$\arg\min_{\boldsymbol{\beta}} \frac{1}{L} \|\mathbf{y} - Z\boldsymbol{\beta}\|_2^2. \tag{5.7}$$

Note that matrix Z in (5.7) can become very tall in cases when there are many data sequences N, each of large length T_i. The standard approaches of estimating $\boldsymbol{\beta}$, based on regular QR decomposition [86], become impractical. For this purpose, in practice, we use the approach of [87] based on Tall and Skinny QR (TSQR), which enables to perform QR of a tall matrix in a block-by-block manner.

To summarize, the execution of the above two steps can be computed in a very efficient, parallel manner. For instance, E step is completely separable across flights and although M step (main computation is (5.7)), requires synchronization, it can also be distributed with the use of parallel QR [87].

5.3 Description of Anomaly Detection Framework

Given a dataset of N *unlabeled* flights our objective is to detect which of them deviate from the normal behavior the most. Since there is no a-priori knowledge on which flights belong to which category, we cannot build a separate model for each class. Our approach is to construct a single SMS-VAR model using *all* flight data and then evaluate the built model on *all* the flights to detect anomalies (see Figure 5.1 for an illustration). Assuming that the fraction of anomalous flights in the dataset is small, we expect that the constructed model mostly represents a typical normal aircraft behavior and is not significantly influenced by the abnormal data. See Section 5.5.1 for further discussion of this point.

Evaluation of the constructed model on the flights is a critical step of the approach since it determines the detection accuracy of the framework. One simple and straightforward choice is the computation of likelihood of each flight. For example, given estimated model parameters Θ, we can compute the likelihood of the data of flight i (dropping i related to the numbering of time series to avoid clutter)

$$\begin{aligned} p(F) &= p(\bar{d}_{1:T}, \bar{m}_{1:T}, \bar{y}_{1:T}) = \\ &= p(\bar{d}_1, \bar{m}_1, \bar{y}_1) \prod_{t=2}^{T} p(\bar{d}_t, \bar{m}_t, \bar{y}_t | \bar{d}_{1:t-1}, \bar{m}_{1:t-1}, \bar{y}_{1:t-1}) \end{aligned} \tag{5.8}$$

where $\ell_t = p(\bar{d}_t, \bar{m}_t, \bar{y}_t | \bar{d}_{1:t-1}, \bar{m}_{1:t-1}, \bar{y}_{1:t-1})$ is the likelihood of observation at t given data seen so far. At each time stamp we can monitor the flight and flag down the times when probability drops to small values, signaling of anomalous activity. However, as we subsequently show in Section 5.5, this metric did not perform well as compared to the approach we propose in this work and discuss below.

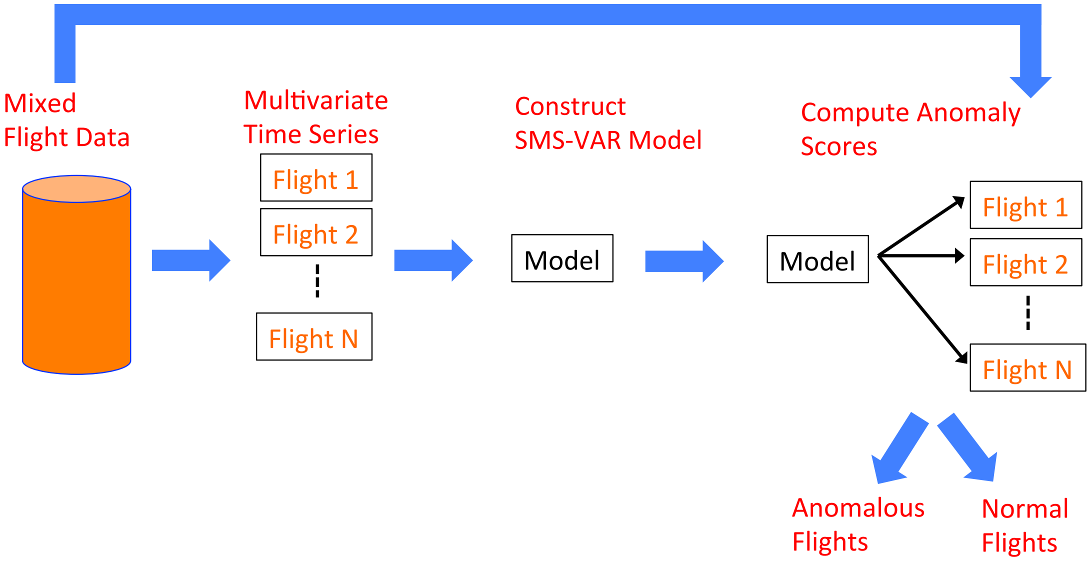


Figure 5.1: Anomaly detection framework using SMS-VAR modeling.

Our proposed method to monitor flight behavior is based on evaluating a discrepancy between one-step ahead predicted and the filtered phase distribution, i.e., after data observation. Specifically, assume that the current phase distribution x_t over $\{1, \ldots, n_x\}$ is $p(x_t|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t})$. Then we can propagate this probability one step forward using estimated model parameters to get prior estimate of phase distribution at $t+1$ (see left plot of Figure 5.1 for details):

$$\begin{aligned} & p(x_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t}) = \\ & = \sum_{d_{t+1}, m_{t+1}, x_t, y_{t+1}} p(d_{t+1}, m_{t+1}, x_t, x_{t+1}, y_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t}) \\ & = \sum_{d_{t+1}, m_{t+1}, x_t, y_{t+1}} p(x_t|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t})\ p(d_{t+1}|m_{t+1}, \bar{d}_t) \times \\ & \times p(m_{t+1}|\bar{m}_t, \bar{d}_t)\ p(x_{t+1}|x_t, m_{t+1}, \bar{d}_t)\ p(y_{t+1}|\bar{y}_t, x_{t+1}). \end{aligned} \tag{5.9}$$

At time $t+1$ we observe data $\bar{m}_{t+1}$, $\bar{d}_{t+1}$, $\bar{y}_{t+1}$, and so the posterior (filtered) distribution of the phase changes to

$$\begin{aligned} p(x_{t+1}|\bar{d}_{1:t+1}, \bar{m}_{1:t+1}, \bar{y}_{1:t+1}) = \\ = \frac{p(\bar{d}_{t+1}, \bar{m}_{t+1}, x_{t+1}, \bar{y}_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t})}{\sum_{x_{t+1}} p(\bar{d}_{t+1}, \bar{m}_{t+1}, x_{t+1}, \bar{y}_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t})}, \end{aligned} \tag{5.10}$$

where we computed

$$\begin{aligned} & p(\bar{d}_{t+1}, \bar{m}_{t+1}, x_{t+1}, \bar{y}_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t}) = \\ & = \sum_{x_t} p(\bar{d}_{t+1}, \bar{m}_{t+1}, x_t, x_{t+1}, \bar{y}_{t+1}|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t}) \\ & = \sum_{x_t} p(x_t|\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t})\ p(\bar{d}_{t+1}|\bar{m}_{t+1}, \bar{d}_t) \times \\ & \times p(\bar{m}_{t+1}|\bar{m}_t, \bar{d}_t)\ p(x_{t+1}|x_t, \bar{m}_{t+1}, \bar{d}_t)\ p(\bar{y}_{t+1}|\bar{y}_t, x_{t+1}). \end{aligned} \tag{5.11}$$

Next, given (5.9) and (5.10), i.e., the distribution of the hidden phase before and after observations at time $t + 1$, we compare these two distributions and measure their difference. For this purpose, we use Kullback–Leibler (KL) divergence, which is defined as

$$\begin{aligned} & D_{t+1}\Big[p(x_{t+1}|F_{1:t})\Big|\Big|p(x_{t+1}|F_{1:t+1})\Big] = \\ & = \sum_{x_{t+1}\in\{1,\dots,n_x\}} p(x_{t+1}|F_{1:t}) \log \frac{p(x_{t+1}|F_{1:t})}{p(x_{t+1}|F_{1:t+1})}, \end{aligned} \tag{5.12}$$

where $F_{1:t}$ is a shorthand for $\{\bar{d}_{1:t}, \bar{m}_{1:t}, \bar{y}_{1:t}\}$, and similarly for $F_{1:t+1}$. Observe from (5.9) and (5.11) that information about all the observed variables, $\bar{d}_t$, $\bar{m}_t$ and $\bar{y}_t$ participate in the computation of the phase distribution. The prior $p(x_{t+1}|F_{1:t})$ reflects the model's belief based on data seen so far about the probability of which phase (i.e., VAR process) currently is active. After data observation, if the posterior $p(x_{t+1}|F_{1:t+1})$ shows a different phase distribution, then the distance measure (5.12) captures this by producing a large D_t. On the other hand, when both distributions are similar, it means the observed data are likely to have been generated from the model and the computed value D_t is small. Moreover, observe that incremental nature of the computation of D_t values implies that our approach can be used for *online* anomaly detection, i.e., algorithm can monitor a flight in real time, without the need to wait for all the data to arrive.

Finally, once we compute D_t's, the anomaly score S, characterizing the flight's abnormality, is obtained by computing a standard deviation (STD) of all D_t, i.e.,

$$S_{\mathrm{KL}} = \frac{1}{T}\sum_{t=1}^{T}(D_t - \mu_D)^2, \tag{5.13}$$

where $\mu_D = \frac{1}{T}\sum_{t=1}^{T} D_t$. Thus, small values of S indicate normal flights, while large S correspond to anomalies.

To motivate our choice for using STD, note that the first-order SMS-VAR is a short-memory model (see Section 5.2), which usually detects short-duration anomalies. Moreover, the values D_t, comparing phases at neighboring times, are sensitive to short-duration anomalies, and so for a typical abnormal flight, D_t usually stay small except for a few time stamps, at which anomaly events occurs and D_t spike. Therefore, STD which uses a sum of quadratic deviations, is sufficiently sensitive to outliers and can serve as an adequate summary measure. We note that detecting abnormal flights based on D_t values can be viewed as a separate outlier detection problem in univariate time-series. Various sophisticated approaches can be used for this, e.g., based on support vector regression [90], mixture transition distribution approach of [91] or a median information from the neighborhood [92], although we observed that a simple standard deviation performed well in practice.

5.4 Compared Algorithms

In this Section we present an overview of the algorithms which we used in the comparison studies in Section 5.5.The five algorithms we considered were: SMS-VAR KL, based on KL divergence (discussed in Section 5.3), SMS-VAR based on log-likelihood, vector autoregressive (VAR) model, semi-Markov switching (SMM) model and the multiple kernel anomaly detector (MKAD).

SMS-VAR LL. This approach is based on estimating SMS-VAR using all the data points and evaluating the flights based on log-likelihood of data. However, instead of using $p(F)$ from (5.8) directly as a measure of abnormality, which usually washes out the events of interest and makes them undetectable; similarly as in (5.13) we used standard deviation, i.e., $S_{\mathrm{LL}} = \frac{1}{T}\sum_{t=2}^{T}(\log \ell_t - \mu_\ell)^2$, where $\ell_t = p(\bar{d}_t, \bar{m}_t, \bar{y}_t | \bar{d}_{1:t-1}, \bar{m}_{1:t-1}, \bar{y}_{1:t-1})$ and μ_ℓ is the mean of ℓ_t's.

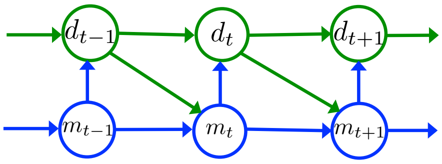
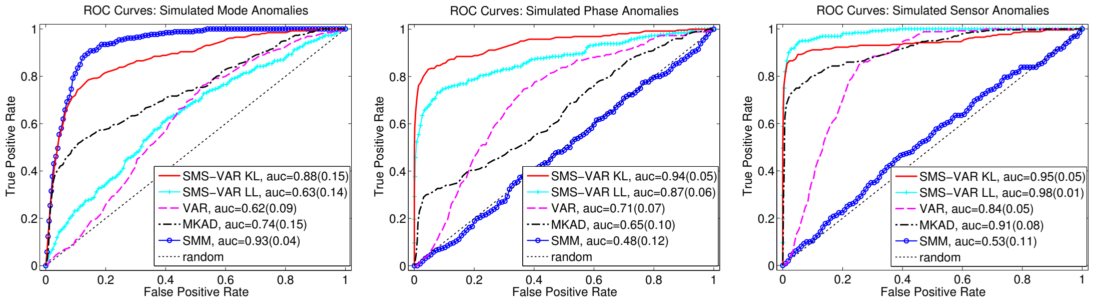


Figure 5.1: Semi-Markov Model.

VAR. In this approach we modeled only the continuous part of the data using a single first-order VAR process $y_t = Ay_{t-1} + \epsilon_t$. The anomaly detection is based on computing $S_{\text{VAR}} = \frac{1}{T}\sum_{t=2}^{T}(r_t - \mu_r)^2$, where $r_t = \|y_t - \hat{A}y_{t-1}\|_2$ is a one-step-ahead prediction error, $\hat{A}$ is the estimated VAR matrix using all the data and μ_r is the mean.

SMM. An approach based on modeling only the discrete part of data is based on semi-Markov model shown in Figure 5.1. Here, similarly as in SMS-VAR, we modeled m_t and d_t using (5.1) and (5.2), respectively. The anomaly score is then computed as $S_{\text{SMM}} = \frac{1}{T}\sum_{t=2}^{T}(\ell_t - \mu_\ell)^2$, where $\ell = p(\bar{d}_t, \bar{m}_t|\bar{d}_{1:t-1}, \bar{m}_{1:t-1})$.

MKAD. This algorithm was designed to detect anomalies in the heterogeneous multivariate time series, where both discrete and continuous features are present. Let F^i and F^j denote the multivariate time series of two flights. The algorithm constructs a kernel of the form $K\Big(F^i, F^j\Big) = \alpha K_d\Big(F^i, F^j\Big) + (1-\alpha)K_c\Big(F^i, F^j\Big)$, where K_d is a kernel over discrete sequences and K_c is a kernel over continuous time series and $\alpha \in [0, 1]$ is a weight, usually set to $\alpha = 0.5$. For discrete sequences, the normalized longest common subsequence (LCS) is used, i.e., $K_d\Big(F^i, F^j\Big) = \frac{|LCS(F^i,F^j)|}{\sqrt{T_iT_j}}$, where $|LCS(F^i, F^j)|$ denotes the length of LCS. For continuous sequences, the kernel $K_c\Big(F^i, F^j\Big)$ is inversely proportional to the distance between symbolic aggregate approximation (SAX) representation [75] of continuous sequences in F^i and F^j. The constructed kernel $K \in \mathbb{R}^{N\times N}$, N is the number of flights, is then used in one-class support vector machine (SVM) to construct a hyperplane to separate anomalous and normal flights. Note that the main idea behind SAX technique is to represent a continuous sequence as a discrete one. This is achieved by dividing a sequence into equally spaced segments, computing the average of each segment and then discretizing the result into a set of alphabets of predefined size. By regulating the length of the segments, MKAD can be tuned to detect long- or short-term dependencies in the data.



a) b) c)

Figure 5.2: Performance of the five algorithms at detecting three simulated types of anomaly events: a) Mode anomaly, b) Phase anomaly, c) Sensor anomaly. There were 100 normal and 10 anomalous flights in each considered scenario. The evaluation was done using the area under ROC curve (AUC). The AUC values are shown after averaging the results over 30 runs, the values in parenthesis show one standard deviation.

5.5 Experimental Results

In this Section we present extensive evaluations of the SMS-VAR model on both synthetic as well as real flight data. We compare the performance of the proposed approach with the four alternatives described in Section 5.4.

5.5.1 Synthetic Data

In the first simulation scenario we study detection accuracy of all the algorithms when presented with different types of anomalies, while in the second study we examine how the proportion of anomalous to normal flights affects the detection accuracy of SMS-VAR algorithm.

Detecting Different Types of Anomalies

We generated three synthetic datasets consisting of 100 normal flights and 10 anomalous, each of length 200 time stamps. The number of phases and sensor measurements in the generation model was set, respectively, to $n_x = 3$ and $n_y = 4$. The number of binary switches was set to 5, which corresponds to $n_m = 2^5 = 32$. In each dataset, the 10 abnormal flights represented a different type of anomaly. In the first dataset, each anomalous flight contained several mode anomalies, i.e., unusual flips of switches, while the continuous sensor measurements behaved normally. In the second dataset,

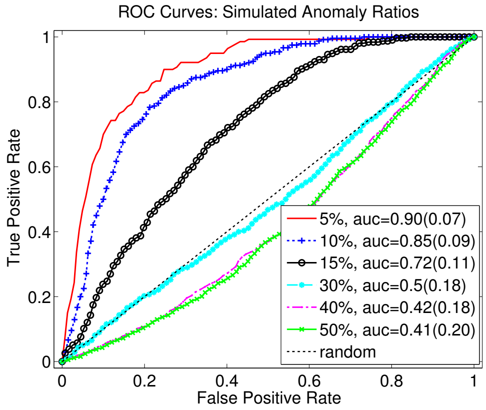


Figure 5.1: Anomaly detection accuracy of SMS-VAR KL when applied to datasets with different proportion of anomalous data ($5\% - 50\%$ of normal data is replaced with anomalous). Total number of flights in all scenarios is 100.

we simulated unusual change of unobserved phases, i.e., at several time stamps one VAR process switched to another VAR process. Note that in this scenario all the measurements related to a particular VAR process were normal except that the change in underlying dynamics was abnormal. Finally, in the third dataset we simulated errors in sensor measurements (continuous data), while the mode transitions behaved normally.

The results are shown in Figure 5.2. It can be seen from Figure 5.2-a that the discrete anomaly detector SMM had the highest accuracy for detecting mode anomalies, followed by SMS-VAR and others. Since the dataset contained only discrete anomalies, it is expected that SMM performed the best. It was also expected that SMM would do poorly on the other two datasets since it could not use the information from sensor measurements. Similarly, the VAR model-based anomaly detector did not do well on the first dataset in Figure 5.2-a but improved its accuracy in Figures 5.2-b, c. MKAD algorithm, using both discrete and continuous data, achieved medium level accuracy in detecting discrete anomalies and did well on the dataset with errors in sensor measurements (we used a SAX window of size 2, since anomalies are of short duration). The algorithm, on the other hand, was insensitive to unusual phase changes in the data.

In contrast, the SMS-VAR-based algorithm, which fuses information from both sources of measurements is able to detect the anomalous flights with high accuracy

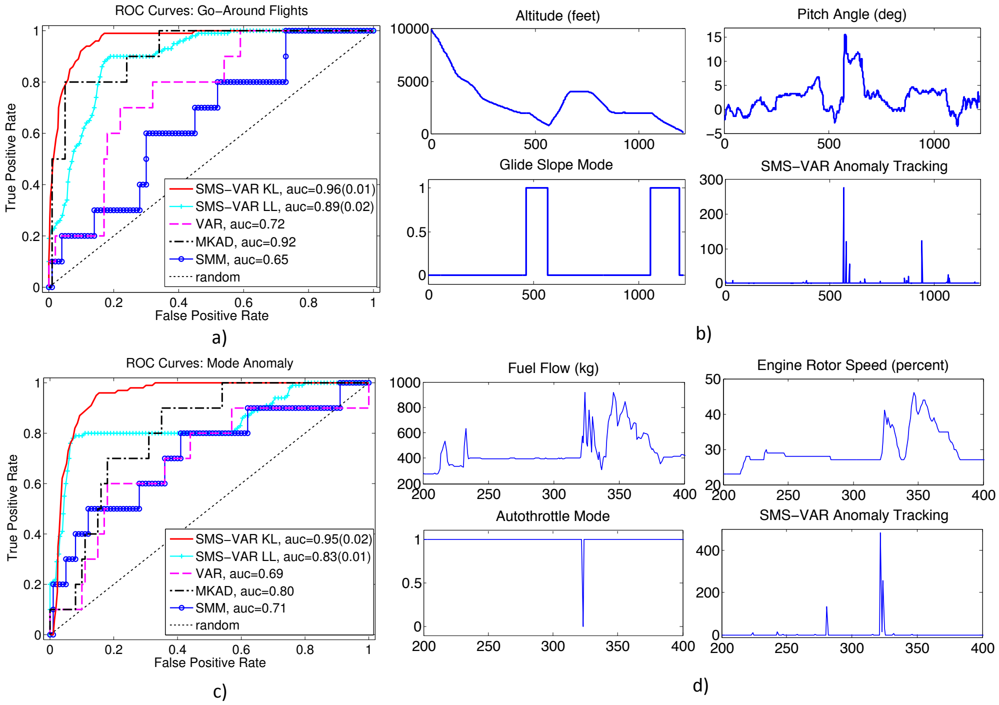


Figure 5.2: Performance of the five approaches at detecting go-around (top) and mode anomaly (bottom) flights. There were 100 regular and 10 anomalous flights in each dataset. Figures a) and c) show ROC curves for each method. AUCs shown after averaging over 30 runs for SMS-VAR, for other methods no averaging is done since they are deterministic. Figure b) shows an example of SMS-VAR KL detecting a typical go-around flight with a history of some of the parameters. Figure d) shows detection of a typical mode anomaly flight.

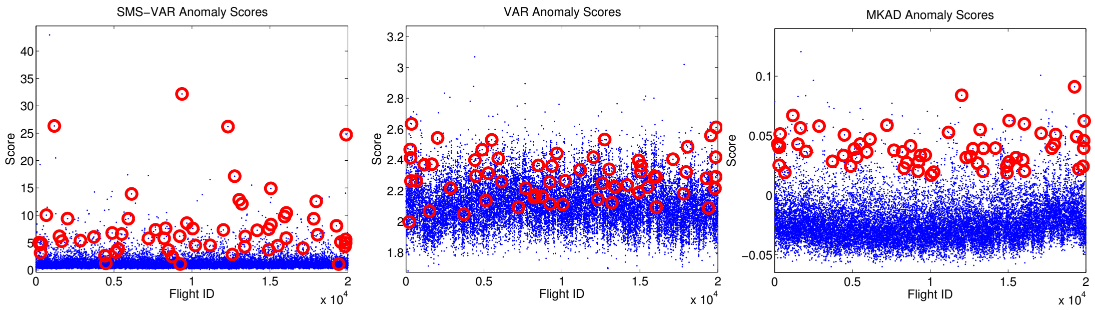
across all the datasets. Moreover, the anomaly detection approach based on KL measure performed better than the log-likelihood measure (LL) on first and second datasets and did slightly worse on the third one, confirming that it is a viable method to detect abnormal activities.

Effect of Anomaly Proportion on Accuracy In this study we investigate the validity of the assumption made in Section 5.3, where we mentioned that for our anomaly detection approach to be accurate, the fraction of anomalous flights should not be large. For this purpose we generated a set of datasets of same size but with different proportion ($5\% - 50\%$) of anomalous sequences. The simulated anomaly was related to unusual switch changes; the parameter dimensions in the generation model remained the same as above. Results are shown in Figure 5.1. It can be seen that when the number of irregular data sequences is small ($5\% - 15\%$), the algorithm's detection accuracy is high (AUC value is $0.9 - 0.7$). On the other hand, as we replace more and more normal flights with anomalous, the accuracy drops. This can be explained by noting that SMS-VAR is built using all the data, so when the fraction of anomalous flights is large, the constructed model represents now a typical *anomalous* behavior. For example, in Figure 5.1 we see that when 50% or more flights are anomalous, the algorithm prediction flips and it starts classifying normal flights as anomalous.

5.5.2 Real Flight Data

In this section we present the evaluation results of the considered approaches on the FOQA flight dataset from a partner airline company (a similar, publicly available flight dataset can be found at [42]). The data contains over a million flights, each having a record of about 300 parameters, including sensor readings, control inputs and weather information, sampled at 1 Hz. Out of 300 parameters, we selected 31 continuous ones related to aircraft's sensor measurements and 18 discrete binary parameters related to pilot switches. Therefore, the model dimensions are $n_y = 18$ and $n_m = 2^{18}$. Note that although 18 binary features correspond potentially to $2^{18} = 262144$ unique combinations, however the number of such values in real data is much smaller, since only few combinations of binary flight switches are possible. For example, in the dataset consisting of 20000 flights, used in Section 5.5.2, only 673 unique modes exist (0.3% of 262144). The number of hidden phases was set to $n_x = 5$, after some preliminary experiments in which we tested several n_x in range $[3, 20]$ and selected n_x balancing a good prediction accuracy with low computational complexity of algorithm. We have selected flights with landings at the same destination airport with aircrafts of the same fleet and type, so that we eliminate potential differences related to aircraft dynamics or landing patterns. Data analysis focused on a portion of the flight below 10000 feet until touchdown (duration 600-1500 time stamps), usually having the highest rates of accidents [95].

The evaluation of the algorithms is first done by using two small dataset of manually labeled flights (Section 5.5.2), while in the second study we use a large dataset in a more



a) b) c)

Figure 5.3: Anomaly scores of SMS-VAR KL, VAR and MKAD on unlabeled dataset consisting of 20000 flights. Red circles in all plots denote all the 61 go-around flights present in the 20000 flights.

SMS-VAR KL	VAR	MKAD
go-around (19) high speed in approach (5) high rate of descent in approach (4) bounced landing (2) delayed braking at landing(2) late retraction of landing gear (4) deviation from glide-slope (2) unusual flight switch changes (11)	go-around (3) fast approach (2) high speed in approach (5) high rate of descent in approach (4) bank cycling in approach(2) high pitch at touch down (1)	go-around (17) high pitch at touch down (1) high speed in approach (2) low speed at touch down (1) low path in approach (1) flaps retracted in approach (1) unusual flight switch changes (15)

Table 5.1: Anomalies discovered in the top 100 anomalous flights, ranked by each anomaly detection method in the set of 20000 unlabeled flights. The distribution of anomaly scores for each method is shown in Figure 5.4.

realistic scenario with no information about flights' labels (Section 5.5.2). In the second case the shown results are only qualitative since no ground truth is available but the discoveries were validated by the domain experts.

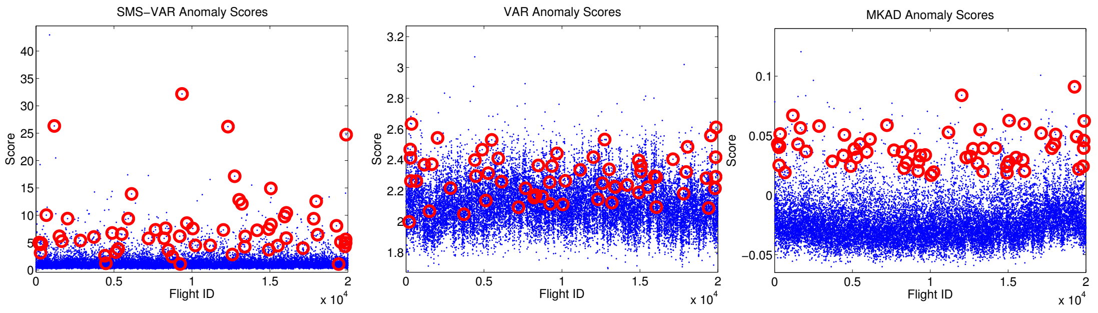
Labeled Flights: Go-Around For this study we manually selected flights, which abort their normal landing, fly back up to a certain altitude and try to repeat the landing again (see Figure 5.1 for an example). These flights are considered operationally significant anomalies since they could be executed in response to an emergency or unsafe conditions in the air or on the runway. The dataset included 10 go-around and 100 normal flights. The results are shown in Figure 5.2. In particular, from Figure 5.2-a we see that SMS-VAR and MKAD performed similar, with SMS-VAR based on KL measure achieving the highest accuracy of detection. Both SMM and VAR-based approaches, which used only part of data, performed worse, missing many go-around flights.

In Figure 5.2-b we also show a typical go-around flight and the corresponding history of D_t (see definition in (5.12)) values across the flight. Note that the go-around happens shortly after $t = 500$, when the altitude increases to about 5000 feet, coupled with unusual behavior in other parameters. For example, a glide slope mode was switched off, which typically is used to safely descend aircraft to a runway. Also, there was a sharp rise of pitch angle, corresponding to airplane's nose lifting up during ascent. The SMS-VAR KL model correctly detected these unusual changes with multiple spikes around the time the go-around was initiated.

Labeled Flights: Mode Anomaly Additionally, we tested the performance of the algorithms on the real flight anomaly related to unusual auto-throttle switchings, whose example is shown in Figure 5.2-d. In particular, around $t = 400$ during aircraft's descent, one of the switches related to throttle control is switched off briefly ($2-5$ seconds). This action led to a fast spool up of engines (from 25% to 40% within few seconds). This abnormal behavior usually causes a quick increase of longitudinal acceleration leading to an abrupt forward motion of the aircraft. Similarly as before, we created a dataset consisting of 10 anomalous and 100 normal flights and the results are shown in Figure 5.2-c.

Interestingly, although the anomaly type was discrete, the simple SMM approach, which looks only at discrete part of data, did not perform well. Similarly, using only continuous features, the VAR algorithm also did poorly. When the information from both sources is combined, as was done in SMS-VAR KL or MKAD, the detection accuracy increased. Still, it can be seen that SMS-VAR KL performed better than other methods by a margin, justifying our proposed approach to track anomalies using the phase information.

Unlabeled Flights Finally, we compared the algorithms' performance on a dataset containing 20000 unlabeled flights. We tested SMS-VAR KL and compared its performance with MKAD and VAR. Figure 5.4 shows the anomaly scores for the three approaches. For each method, we examined the top 100 flights with the highest anomaly scores to determine the flights with operationally significant events. In Table 5.2 we



a) b) c)

Figure 5.4: Anomaly scores of SMS-VAR KL, VAR and MKAD on unlabeled dataset consisting of 20000 flights. Red circles in all plots denote all the 61 go-around flights present in the 20000 flights.

present a summary of the discovered anomalies, examined and validated by the experts. Note that since there is no ground truth available, the presented results are only qualitative.

SMS-VAR KL	VAR	MKAD
go-around (19)	go-around (3)	go-around (17)
high speed in approach (5)	fast approach (2)	high pitch at touch down (1)
high rate of descent in approach (4)	high speed in approach (5)	high speed in approach (2)
bounced landing (2)	high rate of descent in approach (4)	low speed at touch down (1)
delayed braking at landing(2)	bank cycling in approach(2)	low path in approach (1)
late retraction of landing gear (4)	high pitch at touch down (1)	flaps retracted in approach (1)
deviation from glide-slope (2)		unusual flight switch changes (15)
unusual flight switch changes (11)		

Table 5.2: Anomalies discovered in the top 100 anomalous flights, ranked by each anomaly detection method in the set of 20000 unlabeled flights. The distribution of anomaly scores for each method is shown in Figure 5.4.

As can be seen, among the top 100 flights, we found that the most common type of anomaly were the go-around flights, shown as red circles in Figure 5.4, as well as anomalies related to unusual pilot switches. The number of go-arounds detected by SMS-VAR and MKAD was similar, 19 and 17, respectively. The VAR-based approach only identified 3 such flights in its 100 top anomalous flight list. SMS-VAR and MKAD algorithms have additionally identified many flights with unusual changes in pilot switches, although these flights did not overlap. In particular, the anomalies identified by SMS-VAR are characterized by quick changes in the flight parameters (few seconds), e.g., the switchings in auto throttle system as in Figure 5.2-d or the flights when the localizer switch was turned off during approach resulting in large deviation from glide slope and these are difficult to detect using MKAD. The discrete anomalies identified by MKAD are of longer duration. For example, it found several flights when the flight director was switched off for over 2 minutes during the approach. It is an unusual behavior since, typically, flight director is used throughout the approach to assist the pilot with vertical and horizontal cues. Therefore, we can conclude that although achieving better performance in detecting certain types of anomalies, the proposed framework can be positioned as complementary to the existing state-of-the-art approaches, enabling the discoveries of more diverse spectrum of operationally significant events.

Chapter 6

Conclusion

In this thesis, we have presented the work which addresses the problem of anomaly detection in aviation systems. In particular, we have designed a spectral algorithm for learning and inference in hidden semi-Markov models (HSMMs), which were used to model discrete flight data and perform anomaly detection. We have then developed a framework for anomaly detection in the continuous data based on vector autoregressive models (VAR). Moreover, we have presented a theoretical analysis for error bounds and sample complexity in regularized least-squares problems employed for estimating VAR parameters. Finally, we have proposed to combine the HSMM and VAR into a SMS-VAR model to represent a heterogeneous flight data consisting of a mixture of discrete and continuous data.

References

[1] P. J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[2] A. Leroy and P. Rousseeuw. Robust regression and outlier detection. *Wiley Series in Probability and Mathematical Statistics*, 1, 1987.

[3] M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proceedings of the ACM SIGMOD international conference on management of data*, pages 93–104, 2000.

[4] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Data Mining*, pages 226–231. AAAI Press, 1996.

[5] A. Lazarevic, A. Ozgur, L. Ertoz, J. Srivastava, and V. Kumar. A comparative study of anomaly detection schemes in network intrusion detection. In *International Conference on Data Mining*, 2003.

[6] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques.* 2009.

[7] D. Barber. *Bayesian Reasoning and Machine Learning.* Cambridge University Press, 2012.

[8] B. Matthews, S. Das, K. Bhaduri, K. Das, R. Martin, and N. Oza. Discovering anomalous aviation safety events using scalable data mining algorithms. *Journal of Aerospace Information Systems*, 11(7):482, July 2014.

[9] A. Pouliezos and G. S. Stavrakakis. *Real time fault monitoring of industrial processes*, volume 12. Springer Science & Business Media, 2013.

[10] D. Collett. *Modelling survival data in medical research.* CRC press, 2015.

[11] S. Ling, M. McAleer, and H. Tong. Frontiers in time series and financial econometrics: An overview. *Journal of Econometrics*, 189(2):245–250, 2015.

[12] T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 255–270, 1991.

[13] C. P. De Campos and Q. Ji. Efficient structure learning of Bayesian networks using constraints. *The Journal of Machine Learning Research*, 12:663–689, 2011.

[14] P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, prediction, and search.* MIT Press, 2000.

[15] S. Yaramakala and D. Margaritis. Speculative Markov blanket discovery for optimal feature selection. In *IEEE International Conference on Data mining*, pages 809–812, 2005.

[16] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation. *The Journal of Machine Learning Research*, 11:171–234, 2010.

[17] I. Tsamardinos, L. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.

[18] 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*, pages 1–38, 1977.

[19] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer-Verlag New York, 2006.

[20] F. Jensen, F. V. Jensen, and S. L. Dittmer. From influence diagrams to junction trees. In *Proceedings of the international conference on Uncertainty in artificial intelligence*, pages 367–373, 1994.

[21] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[23] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[24] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741, 1984.

[25] R. M Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003.

[26] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[27] S. Basu and G. Michailidis. Estimation in high-dimensional vector autoregressive models. *ArXiv e-prints, arXiv:1311.4175*, 2015.

[28] K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning.* PhD thesis, University of California, Berkeley, 2002.

[29] I. Melnyk and A. Banerjee. A spectral algorithm for inference in hidden semi-Markov models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 690–698, 2015.

[30] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, pages 267–288, 1996.

[31] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society.*, 68(1):49–67, 2006.

[32] M. Bogdan, E Berg, W. Su, and E. Candes. Statistical estimation and testing via the sorted l1 norm. *arXiv preprint arXiv:1310.1969*, 2013.

[33] I. Melnyk and A. Banerjee. Estimating structured vector autoregressive models. In *Proceedings of the International Conference on Machine Learning*, 2016.

[34] I. Melnyk, P. Yadav, M. Steinbach, J. Srivastava, V. Kumar, and A. Banerjee. Detection of precursors to aviation safety incidents due to human factors. In *Workshop on Domain Driven Data Mining (in conjunction with ICDM 2013)*, pages 407–412, 2013.

[35] I. Melnyk, B. Matthews, H. Valizadegan, A. Banerjee, and N. Oza. Vector autoregressive model-based anomaly detection in aviation systems. *Journal of Aerospace Information Systems*, 13(4):161–173, 05 2016.

[36] I. Melnyk, B. Matthews, A. Banerjee, and N. Oza. Semi-Markov switching vector autoregressive model-based anomaly detection in aviation systems. *Data Mining and Knowledge Discovery*, 2016.

[37] Federal Aviation Administration. FAA Terminal area forecast summary report. Available at http://www.faa.gov/about/office_org/headquarters_offices/apl/aviation_forecasts/taf_reports/media/TAF_Summary_Report_FY2013-2040.pdf, 2013.

[38] Joint Planning and Development Office. Concept of operations for the Next Generation Air Transportation System. Available at http://www.dtic.mil/dtic/tr/fulltext/u2/a535795.pdf., 2011.

[39] Federal Aviation Administration. Next Generation Air Transportation System. Available at http://www.faa.gov/nextgen/., 2014.

[40] I. C. Statler and D. A. Maluf. NASA's aviation system monitoring and modeling project. Technical report, NASA/TP–2007–214556, 2003.

[41] R. Nehl and J. Schade. Update: Concept and operation of the performance data analysis and reporting system (PDARS). In *Aerospace Conference*, pages 1–16, March 2007.

[42] NASA Flight Dataset. Available at https://c3.nasa.gov/dashlink/projects/85/.

[43] S. A. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of computer security*, 6(3):151–180, 1998.

[44] A. Srivastava, A. Kundu, S. Sural, and A. K. Majumdar. Credit card fraud detection using hidden Markov model. *Dependable and Secure Computing, IEEE Transactions on*, 5(1):37–48, 2008.

[45] W. Wong, A. W. Moore, G. F. Cooper, and M. M. Wagner. Bayesian network anomaly pattern detection for disease outbreaks. In *International Conference on Machine Learning*, pages 808–815, 2003.

[46] T. van Kasteren, G. Englebienne, and B. Kröse. Activity recognition using semi-markov models on real world smart home datasets. *Journal of ambient intelligence and smart environments*, 2(3):311–325, 2010.

[47] H. Zen, K. Tokuda, T. Masuko, T. Kobayasih, and T. Kitamura. A hidden semi-markov model-based speech synthesis system. *Transactions of Information Systems*, E90-D(5):825–834, 2007.

[48] Y. Xie and S.-Z. Yu. A large-scale hidden semi-markov model for anomaly detection on user browsing behaviors. *Transactions on Networking*, 17(1):54–65, 2009.

[49] R. S Tsay. *Analysis of financial time series*, volume 543. 2005.

[50] L. Ljung. *System identification: theory for the user*. Springer, 1998.

[51] P. A. Valdes-Sosa, J. M. Sanchez-Bornot, A. Lage-Castellanos, M. Vega-Hernandez, J. Bosch-Bayard, L. Melie-Garcia, and E. Canales-Rodriguez. Estimating brain functional connectivity with sparse multivariate autoregression. *Philosophical Transactions of the Royal Society*, 360(1457):969–981, 2005.

[52] C. Sims and T. Zha. Were there regime switches in US monetary policy? *The American Economic Review*, pages 54–81, 2006.

[53] L. H. Lehman, S. Nemati, R. P. Adams, G. Moody, A. Malhotra, and R. G. Mark. Tracking progression of patient state of health in critical care using inferred shared dynamics in physiological time series. In *Annual International Conference of the Engineering in Medicine and Biology Society*, pages 7072–7075, 2013.

[54] Y. Bar-Shalom and X.-R. Li. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, 2001.

[55] K. P. Murphy. Hidden semi-Markov models. Available at http://www.cs.ubc.ca/ murphyk/Papers/segment.pdf. 2002.

[56] X. Tan and H. Xi. Hidden semi-Markov model for anomaly detection. *Applied Mathematics and Computation*, 205(2):562 – 567, 2008.

[57] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[58] S. Budalakoti, A. N. Srivastava, and M. E. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(1):101–113, Jan 2009.

[59] M. Saada and Q. Meng. An efficient algorithm for anomaly detection in a flight system using dynamic bayesian networks. In *Proceedings of the 19th International Conference on Neural Information Processing*, pages 620–628, 2012.

[60] A. N. Srivastava. Discovering system health anomalies using data mining techniques. In *Proceedings of Joint Army Navy NASA Airforce Conference on Propulsion*, 2005.

[61] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, 1966.

[62] S.-Z. Yu and H. Kobayashi. An efficient forward-backward algorithm for an explicit-duration hidden Markov model. *IEEE Signal Processing Letters*, 10(1):11–14, 2003.

[63] A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor spectral approach to learning mixed membership community models. In *Conference on Learning Theory*, 2013.

[64] A. Anandkumar, A. Javanmard, D. Hsu, and S. M. Kakade. Learning linear Bayesian networks with latent variables. In *Proceedings of the International Conference on Machine Learning*, volume 28, pages 249–257, 2013.

[65] D. Hsu, S. M. Kakade, and T. Zhang. A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460 – 1480, 2012.

[66] A. Parikh, L. Song, and E. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1065–1072, 2011.

[67] A. Parikh, L. Song, M. Ishteva, G. Teodoru, and E. Xing. A spectral algorithm for latent junction trees. In *Proceedings of the 28th Conference Annual Conference on Uncertainty in Artificial Intelligence*, pages 675–684, 2012.

[68] H. A. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of chemometrics*, 14(3):105–122, 2000.

[69] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[70] D. A. Levin, Y. Peres, and E. Wilmer. *Markov chains and mixing times.* American Mathematical Society, 2009.

[71] A. Anandkumar, D. Hsu, M. Janzamin, and S. M. Kakade. When are overcomplete topic models identifiable? Uniqueness of tensor tucker decompositions with structured sparsity. In *Advances in Neural Information Processing Systems*, pages 1986–1994, 2013.

[72] Dimitry G., Bryan M., and Rodney M. Aircraft anomaly detection using performance models trained on fleet data. In *Conference on intelligent data understanding*, pages 17–23, 2012.

[73] A. N. Srivastava. Greener aviation with virtual sensors: a case study. *Data Mining and Knowledge Discovery*, 24(2):443–471, 2012.

[74] S. Das, B. Matthews, A. Srivastava, and N. Oza. Multiple kernel learning for heterogeneous anomaly detection: Algorithm and aviation safety case study. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 47–56, 2010.

[75] P. Patel, E. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. In *IEEE International Conference on Data Mining*, pages 370–377, 2002.

[76] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[77] Padhraic S. Clustering sequences with hidden Markov models. In *Advances in Neural Information Processing Systems*, pages 648–654, 1997.

[78] A. Panuccio, M. Bicego, and V. Murino. A hidden Markov model-based approach to sequential data clustering. In *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396, pages 734–743. 2002.

[79] H. Lutkepohl. *New Introduction to Multiple Time Series Analysis*. Springer Publishing Company, Incorporated, 2007.

[80] D. N. Gujarati. *Basic econometrics*. Tata McGraw-Hill Education, 2012.

[81] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15, 2009.

[82] L. Bergroth, H. Hakonen, and T. Raita. A survey of longest common subsequence algorithms. In *International Symposium on String Processing and Information Retrieval*, pages 39–48, 2000.

[83] Roger A Horn and Charles R Johnson. *Matrix analysis.* Cambridge university press, 2012.

[84] P. W. Holland and R. E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics - Theory and Methods*, 6(9):813–827, 1977.

[85] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction.* Springer, 2 edition, 2009.

[86] G. H. Golub and C. F. Van Loan. *Matrix computations.* John Hopkins University Press, 3 edition, 2012.

[87] J. Demmel, L. Grigori, M. Hoemmen, and J. Langou. Communication-optimal parallel and sequential QR and LU factorizations. *SIAM Journal on Scientific Computing*, 34(1):A206–A239, 2012.

[88] P. Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

[89] P. C. Mahalanobis. On the generalised distance in statistics. In *National Institute of Science*, pages 49–55, 1936.

[90] J. Ma and S. Perkins. Online novelty detection on temporal sequences. In *International Conference on Knowledge Discovery and Data Mining*, pages 613–618, 2003.

[91] N. D. Le, R. D. Martin, and A. E. Raftery. Modeling flat stretches, bursts outliers in time series using mixture transition distribution models. *Journal of the American Statistical Association*, 91(436):1504–1515, 1996.

[92] S. Basu and M. Meckesheimer. Automatic outlier detection for time series: an application to sensor data. *Knowledge and Information Systems*, 11(2):137–154, 2007.

[93] Meinard Müller. *Information Retrieval for Music and Motion.* Springer-Verlag, 2007.

[94] G. A. Ten Holt, M. J. T. Reinders, and E. A. Hendriks. Multi-dimensional dynamic time warping for gesture recognition. In *Conference of the Advanced School for Computing and Imaging*, 2007.

[95] Boeing. Statistical summary of commercial jet airplane accidents. Available at http://www.boeing.com/news/techissues/pdf/statsum.pdf., 2013.

[96] T. Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[97] G. Box and G. Jenkins. *Time Series Analysis, Forecasting and Control.* Prentice Hall PTR, 3rd edition, 1994.

[98] G. Raskutti, M. J Wainwright, and B. Yu. Restricted eigenvalue properties for correlated Gaussian designs. *The Journal of Machine Learning Research*, 11:2241–2259, 2010.

[99] M. Rudelson and S. Zhou. Reconstruction from anisotropic random measurements. *IEEE Transactions on Information Theory*, 59(6):3434–3447, 2013.

[100] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[101] F. Han and H. Liu. A direct estimation of high dimensional stationary vector autoregressions. *ArXiv e-prints, arXiv:1307.0293*, 2013.

[102] E. Candes and T. Tao. The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

[103] S. Song and P. J. Bickel. Large vector auto regressions. *ArXiv e-prints, arXiv:1106.3915*, 2011.

[104] A. B. Kock and L. Callot. Oracle inequalities for high dimensional vector autoregressions. *Journal of Econometrics*, 186(2):325–344, 2015.

[105] P.-L. Loh and M. J Wainwright. High-dimensional regression with noisy and missing data: Provable guarantees with non-convexity. In *Advances in Neural Information Processing Systems*, pages 2726–2734, 2011.

[106] S. Basu and G. Michailidis. Regularized estimation in sparse high-dimensional time series models. *The Annals of Statistics*, 43(4):1535–1567, 08 2015.

[107] S. Negahban, B. Yu, M. J. Wainwright, and P. K. Ravikumar. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009.

[108] S. Chatterjee, K. Steinhaeuser, A. Banerjee, S. Chatterjee, and G. Ganguly. Sparse group Lasso: Consistency and climate applications. In *Proceedings of International Conference on Data Mining*, pages 47–58, 2012.

[109] A. Banerjee, S. Chen, F. Fazayeli, and V. Sivakumar. Estimation with norm regularization. In *Advances in Neural Information Processing Systems*, pages 1556–1564, 2014.

[110] M. Figueiredo and R. Nowak. Sparse estimation with strongly correlated variables using ordered weighted l1 regularization. *arXiv preprint arXiv:1409.4005*, 2014.

[111] J. Zhou, J. Liu, V. A Narayan, and J. Ye. Modeling disease progression via fused sparse group Lasso. In *Proceedings of International conference on Knowledge discovery and data mining*, pages 1095–1103, 2012.

[112] T. Yang, J. Wang, Q. Sun, D. P. Hibar, N. Jahanshad, L. Liu, Y. Wang, L. Zhan, P. Thompson, and J. Ye. Detecting genetic risk factors for Alzheimer's disease in whole genome sequence data via Lasso screening. In *IEEE International Symposium on Biomedical Imaging*, 2015.

[113] M. Talagrand. *The generic chaining: upper and lower bounds of stochastic processes*. Springer, 2006.

[114] M. Ledoux and M. Talagrand. *Probability in Banach Spaces: isoperimetry and processes*. Springer, 2011.

[115] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S Willsky. The convex geometry of linear inverse problems. *Foundations of Computational mathematics*, 12(6):805–849, 2012.

[116] S. Chen and A. Banerjee. Structured estimation with atomic norms: General bounds and applications. In *Advances in Neural Information Processing Systems*, pages 2890–2898, 2015.

[117] P. Zhao and B. Yu. On model selection consistency of Lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

[118] M. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, 55(5):2183–2202, 2009.

[119] N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *The Annals of Statistics*, pages 246–270, 2009.

[120] A. Argyriou, R. Foygel, and N. Srebro. Sparse prediction with the k-support norm. In *Advances in Neural Information Processing Systems*, pages 1457–1465, 2012.

[121] J. Gutierrez-Gutierrez and P. M. Crespo. Block Toeplitz matrices: asymptotic results and applications. *Foundations and Trends in Communications and Information Theory*, 8(3):179–257, 2011.

[122] M. B. Priestley. *Spectral analysis and time series.* Academic press, 1981.

[123] P. J. Bickel, Y. Ritov, and A. B. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 08 2009.

[124] O. Shamir. A variant of Azuma's inequality for martingales with subgaussian tails. *arXiv preprint arXiv:1110.2392*, 2011.

[125] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *ArXiv e-prints, arXiv:1011.3027*, 2010.

[126] K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19(3):357–367, 1967.

[127] M. Talagrand. *The Generic Chaining.* Springer Monographs in Mathematics. Springer Berlin, 2005.

[128] M. Talagrand. Majorizing measures without measures. *Annals of probability*, pages 411–417, 2001.

[129] S. Negahban and M. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *The Annals of Statistics*, 39(2):1069–1097, 2011.

[130] H. M. Krolzig. Markov-switching vector autoregressions. *Lecture Notes in Economics and Mathematical Systems*, 454, 1997.

[131] R. H. Shumway and D. S. Stoffer. Dynamic linear models with switching. *Journal of the American Statistical Association*, 86(415):763–769, 1991.

[132] Chang-Jin K. Dynamic linear models with markov-switching. *Journal of Econometrics*, 60(12):1 – 22, 1994.

[133] S. Lee, I. Hwang, and K. Leiden. Flight deck human-automation issue detection via intent inference. In *International Conference on Human-Computer Interaction in Aerospace*, 2014.

[134] L. Li, S. Das, H. John, R. Palacios, and A. Srivastava. Analysis of flight data using clustering techniques for detecting abnormal operations. *Journal of Aerospace Information Systems*, 12(9):587–598, 2015.

[135] S. Das, L. Li, A. N. Srivastava, and R. J. Hansman. Comparison of algorithms for anomaly detection in flight recorder data of airline operations. In *AIAA Aviation Technology, Integration, and Operations Conference*, 2012.

[136] J. Janssen and N. Limnios. *Semi-Markov models and applications.* Springer Science & Business Media, 2013.

[137] S. M. Ross. *Introduction to probability models.* Academic press, 2014.

[138] J. D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica: Journal of the Econometric Society*, pages 357–384, 1989.

[139] Z. Ghahramani and G. E. Hinton. Variational learning for switching state-space models. *Neural computation*, 12(4):831–864, 2000.

[140] U. Hassler. *Stochastic Processes and Calculus: An Elementary Introduction with Applications*. Springer, 2015.