RETROSPECTIVE AND EXPLORATORY ANALYSES FOR ENHANCING THE SAFETY OF ROTORCRAFT OPERATIONS

A Dissertation
Presented to
The Academic Faculty

By

Hsiang-Jui Chin

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology

December 2021

RETROSPECTIVE AND EXPLORATORY ANALYSES FOR ENHANCING THE SAFETY OF ROTORCRAFT OPERATIONS

Approved by:

Dr. Dimitri Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Daniel P Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Alexia P Payan
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Jonnalagadda V R Prasad
School of Aerospace Engineering
Georgia Institute of Technology

Mr. Charles Cliff Johnson
William J. Hughes Technical Center
Federal Aviation Association

Date Approved: December 7, 2021

ACKNOWLEDGEMENTS

The work presented in this thesis cannot be accomplished without the assistance and support from many people. I would like to first express my sincere thanks to my committee members for their continued support and guidance. My advisor, Dr. Mavris, has taught me how to approach a problem and connect the dots, and I am genuinely grateful for his unwavering support. Mr. Johnson always had a profound belief in my work and encouraged me to take it to the next level. I really appreciate having the chance to participate in the rotorcraft project that he directed over the years. Dr. Prasad and Dr. Schrage have extensive knowledge in rotorcraft studies, and I would like to express my thanks for their insightful suggestions. Dr. Payan provided numerous advice and encouragement throughout the duration of my work, and I would also like to acknowledge her efforts. Special thanks to Dr. Joseph for valuable discussions on the topic of surrogate modeling, and his helpful advice played an essential role in my work. In the Aerospace System Design Laboratory, there are many friends and colleagues that I learned from and interacted with during my study. It is truly a valuable and memorable experience to work with Joseph Robinson, Paola Zanella, Po-Nien Lin, Zhenyu Gao, Caleb Harris, Etienne Bouchard, Robert Walters, and many others. Lastly, I would like to extend my gratitude to my wife Ying and my family for their continuing support and absolute faith in my ability to finish the work. It goes without saying that their encouragements are the key motivation to keep me going and make this work possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Helicopters are versatile aerial vehicles and are ideal for a variety of operations. However, recent studies have shown that the number and rate of accidents associated with helicopters are increasing. In the 2017 most wanted list, the National Transportation Safety Board (NTSB) recommended expanding the use of flight data recorders in helicopters. This action means that the amount of flight data for routine helicopter operations is expected to proliferate in the coming years, and it is crucial to utilize the information appropriately for knowledge discovery or anomaly detection. Data mining techniques are widely used in the commercial fixed-wing aviation domain to improve operational safety retrospectively. However, studies on anomaly detection for flight data in the rotorcraft domain are not as prevalent, potentially due to the lack of installed flight data recorders and vague definitions of phases of flight. One of the objectives of this research is to develop a framework for improving flight safety specific to rotorcraft operation by retrospectively discovering potential anomalies in flight data records. To pave the way for the task of anomaly detection, we first focus on phases of flight identification, and several methods are proposed to find the homogeneous flight segments. A few flight samples with pre-labeled flight phases were used to compare these techniques, and results show that a regression-based method and a filtering approach can identify flight phases in different altitude regions.

Additionally, exceedance analyses are typically used in flight data monitoring (FDM) for anomaly detection. However, they usually rely on pre-defined thresholds, which might vary depending on the type of operations or the vehicles considered. Without defining thresholds in advance, we proposed a sequential approach that contains three modules for detecting different levels of anomalies. To ensure the effectiveness of the methods selected, synthetic and simulated data are used to compare the performance of the proposed techniques before applying them to the actual flight segments. Then, specific groups of initial climb and the approach segments from a real dataset are used to demonstrate the va-

lidity of the methods chosen in this study. Our test indicates that using functional principal component analysis (FPCA) or a convolutional variational autoencoder (CVAE) can extract shape features in time-series signals in a parsimonious fashion. Along with a density-based clustering method (DBSCAN), we can identify the shape anomalies in flight parameters. Although the detected anomalies might not directly be associated with hazardous events, it may be helpful to assist helicopter operators in discovering patterns not conforming to their standard operating procedures or that do not follow normal operations.

The exploratory analysis aims to develop an efficient methodology to explore the safety envelope of some flight maneuvers and to acquire recovery trajectories for hazardous events. The autorotation maneuver is selected as a use case due to its time criticality and low occurrences in routine helicopter operations. To facilitate the process of predicting the responses, surrogate modeling is used in this study. With this implementation, the responses can be predicted without going through the optimization. Among all the types of surrogates tested in this study, Gaussian process regression with MaxPro design is an adequate method because it can capture the responses of some known functions in both the low or the high-dimensional spaces. To predict the required controls for an unobserved condition in the operational space, we proposed a surrogate that contains two Gaussian processes for handling functional responses in unequal lengths. Finally, a sensitivity analysis is conducted to identify the key parameters that affect the shape of the safety envelope.

CHAPTER 1
INTRODUCTION

1.1 Background

Helicopters are versatile aerial vehicles and they are ideal for a wide variety of operations, such as search and rescue, sight-seeing, and oil rig support. The rotary-wing configuration, which is fundamentally different from fixed-wing aircraft, allows helicopters to have the capability to take off vertically and hover in the air. The superior maneuverability of helicopters relies on a good rotor system design, which was a challenging task at an early development stage. It took thirty more years to demonstrate the first practical helicopter flight compared with the Wright brother's success in 1908. Over the years, engineers have come up with different design variants of helicopters ranging from the ones with tandem rotors, tilt rotors to coaxial rotors and have also introduced several features to improve flight safety, such as using modern avionics onboard for enhancing the situation awareness and employing a wire cutter for preventing wire strike incidents. Helicopter flights are supposed to be more reliable in recent days than in the past; however, people still have doubts about the safety of rotorcraft operations. According to [1], helicopter flights are 85 times more dangerous than automobile transportation. From a recent safety report by the United States Helicopter Safety Team (USHST) [2], it is shown in Figure 1.1 that the fatal accident rate has an increasing trend. In order to reach the goal of 20% reduction of fatal accident rate from the baseline, more attention needs to be directed to the safety of rotorcraft operations.

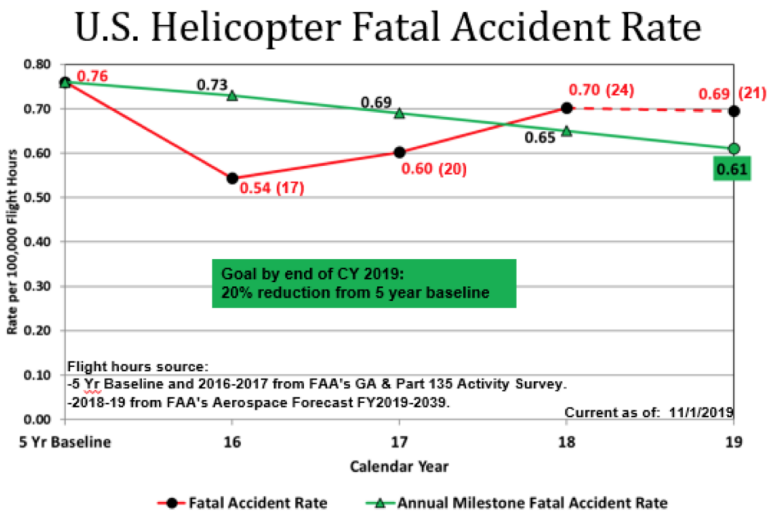


Figure 1.1: Helicopter accident statistics from USHST [2]

To understand the significant events that have happened in rotorcraft accidents, Rao et al. [3] used a state-based accident model to analyze 5051 accidents in the National Transportation Safety Board (NTSB) database from 1982 to 2008. In Figure 1.2, which is drawn based on the data provided in [3], more than half of the fatal accidents for helicopters terminated in a state relevant to inflight collisions with either terrain/water or an object. We can also observe that a small proportion of these accidents do not associate with a recognizable end state, potentially due to damaged recorders or no recorder on board. In Figure 1.3a, hazardous events for both fatal and non-fatal accidents are identified, and the top two on the chart are the loss of control (LOC) and the controlled flight into terrain (CFIT). By looking at the likelihood versus consequence plot in Figure 1.3b, these two hazardous states are situated at the top right corner; therefore, their significance in terms of the occurrence and the severity cannot be ignored. To further dive into what are the triggers for these hazardous events, all precursory events for the LOC and the CFIT are charted separately in Figure 1.4 and 1.5 along with the likelihood versus consequence plots. Obviously, the inflight planning/decision and the remedial actions are the top triggers for these hazardous events. Thus, decision-making is essential for survival in such hazardous events, especially for those that are time-critical.

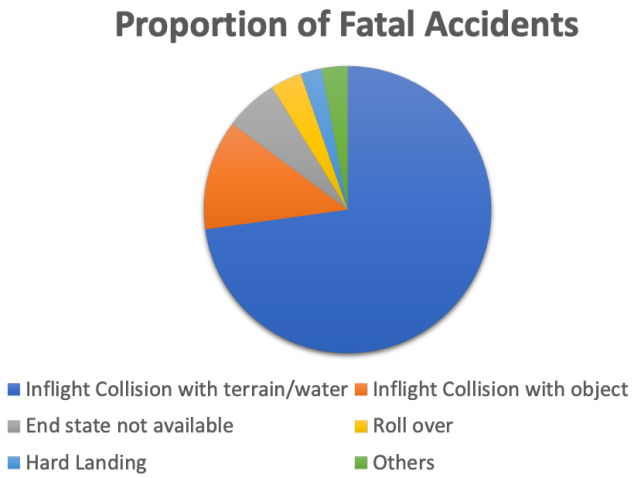
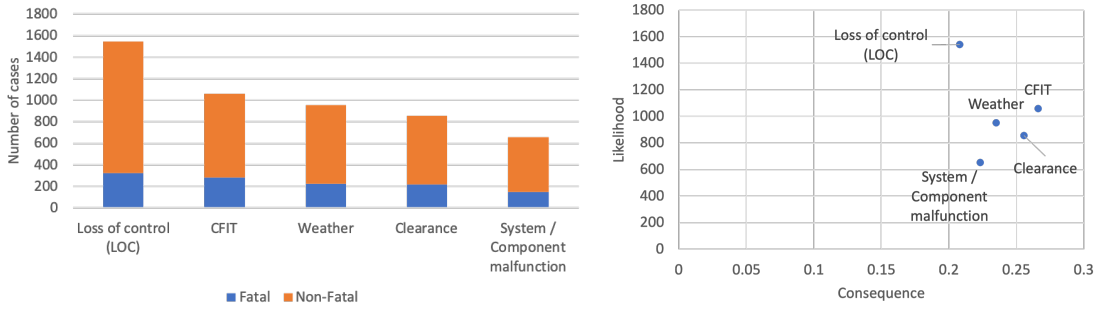


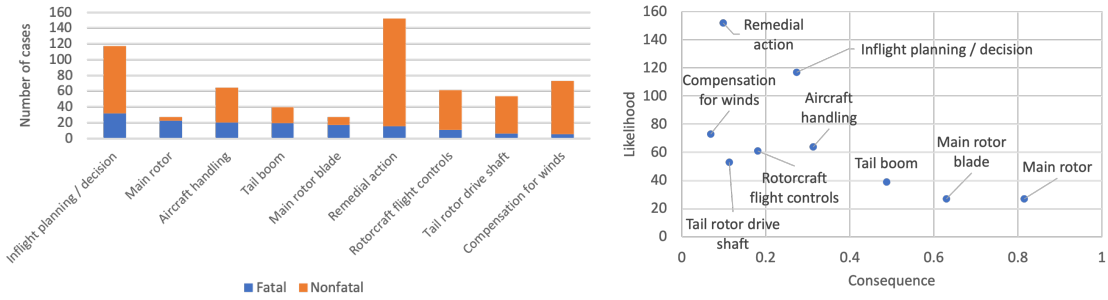
Figure 1.2: Terminal states for fatal accidents (data from [3])



(a) Number of cases for hazardous events

(b) Likelihood versus consequence plot

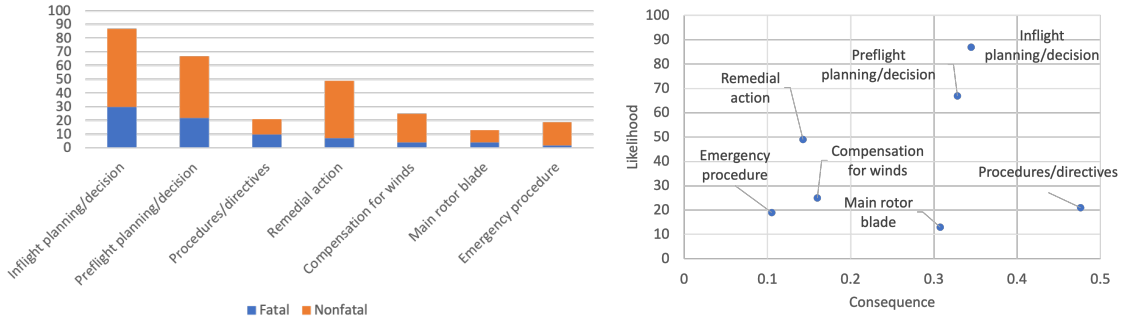
Figure 1.3: Hazardous events for both fatal and non-fatal accidents (data from [3])



(a) Number of cases for the triggers of the LOC

(b) Likelihood versus consequence plot

Figure 1.4: Top triggers for the LOC on both fatal and non-fatal accidents (data from [3])



(a) Number of cases for the triggers of the CFIT

(b) Likelihood versus consequence plot

Figure 1.5: Top triggers for the CFIT on both fatal and non-fatal accidents (data from [3])

To look at the helicopter accidents from another perspective, the U.S. Joint Helicopter Safety Analysis Team (JHSAT) [4] provided a compendium report in which 523 helicopter accidents occurred in the early 2000s were investigated. In Figure 1.6, the accidents are grouped by the occurrence category, and the top two on the chart are the LOC and the autorotation. Another way to categorize these accidents is to examine what phases of flight they are associated with. As shown in Figure 1.7, the majority of accidents, including both fatal and non-fatal ones, are associated with the landing phase. If we only considered the fatal accidents, the en-route phase would be the one to have the most prominent case number. Given the availability of accident statistics, researchers can have a better picture of what happened during accidents and are also able to identify prevalent patterns in multiple accidents.

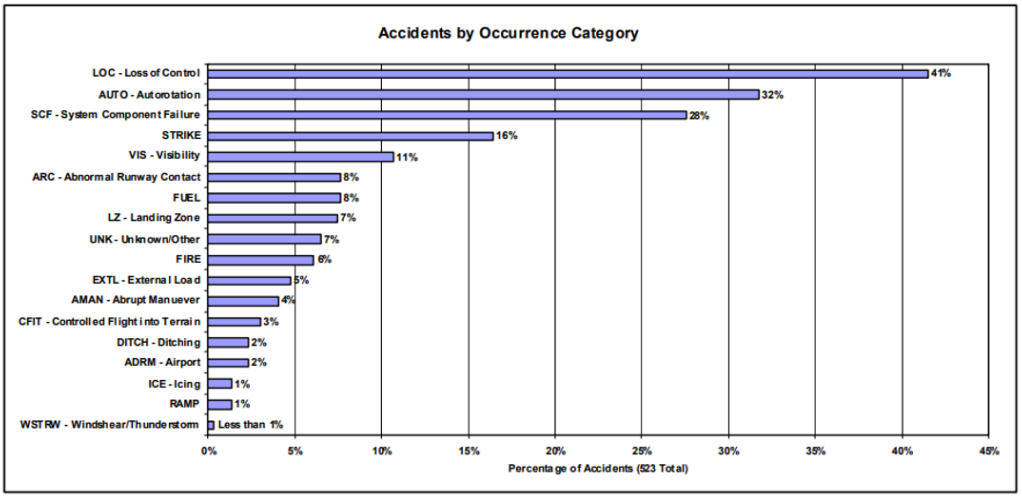


Figure 1.6: Helicopter accidents by occurrence category [4]

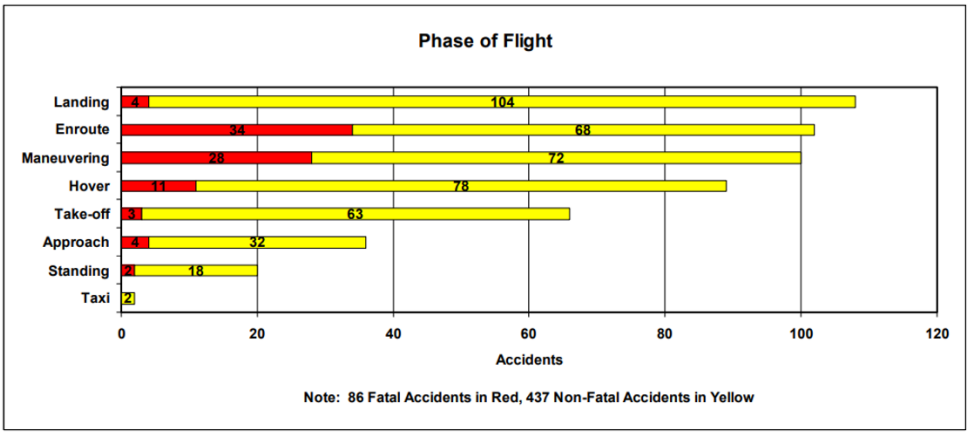


Figure 1.7: Helicopter accidents grouped by phases of flight [4]

To determine the root cause of aviation accidents, investigators typically use the 5M model [5] for accident analysis. The 5M model contains five ingredients: man, machine, medium (environment), mission, and management. To look at the accidents from the perspective of "man" in 5M, pilots who had less than 500 flying hours are accounted for 45% of the accidents among all cases considered in [4]. The issue of immature pilots can be remedied through flight training programs. With more experience gained from either actual flights or simulators, the chances of making good pilot decisions can be potentially increased. When looking from the perspective of "machine" in 5M, 28% of the accidents

can be attributed to system component failure as shown in Figure 1.6. To enhance the reliability of parts in a helicopter, sensors can be placed inside the vehicle for health monitoring and with the information acquired, a comprehensive maintenance schedule can be established. For the third "M" medium, inferior operating conditions like bad weather or proximity to obstacles are likely to increase the risk of accidents. On a foggy day, pilots need to pay additional attention to the change of weather, which tends to increase their workload. If the weather information can be predicted and be fed to pilots promptly, they may be better aware of the situation; thus, this type of risk can then be mitigated. Similar to the medium, high-risk missions may drastically increase the workload for pilots. An example of high-risk missions would be an emergency medical service (EMS) type of mission that is usually planned quickly and has inadvertent events during the operation. The last "M" refers to the management and it is the overarching element of the 5M model. According to [5], the management dictates the distribution of resources, and the way resources being allocated would determine its success or failure. Suppose an operator runs the business in a mountainous area and the management decided not to purchase terrain warning systems for its helicopters. On those days with low visibility conditions, the operations would have higher risks than those on clear days.

A safety management system (SMS) [6] is the standard method for mitigating the risk of aerial operations. There are four pillars in the SMS: the safety policy, safety risk management, safety promotion, and safety assurance. We will focus on safety assurance because some activities, including data analysis and hazard identification, are directly related to flight data monitoring (FDM), which is also an approach for risk mitigation. Based on the Civil Aviation Authority (CAA), FDM is a "systematic method of accessing, analyzing, and acting upon the information obtained from digital flight data records of routine flight operations to improve safety. It is a proactive and timely use of flight data to identify and address operational risks before they can lead to incidents and accidents." The Federal Aviation Administration (FAA) defines FDM as "the technology and methodology for

collecting and analyzing data recorded in flight." According to these quotes, three major components of an FDM program are described as the following:

- Data collection and acquisition
- Flight data analysis
- Data visualization and remedial actions

A detailed process flow of a typical FDM program is extracted from [7] and it is shown in Figure 1.8. The feedback mechanism is essential to improve the safety of operations in a retrospective manner. Two programs for helicopter operations in general aviation are relevant to FDM: Helicopter Flight Data Monitoring (HFDM) and Health and Usage Monitoring System (HUMS). In the HFDM, flight data analysis primarily focuses on detecting exceedances and safety-related events. Exceedances are limits or thresholds placed on flight parameters to detect abnormal timestamps in the flight data records. Manufacturers and operators typically determine these values, and they may not stay the same for various types of missions. Moreover, events are usually defined through multiple flight parameters over a period of time. Sometimes, due to the inaccessibility of certain parameters in less-capable recorders, inferences need to be made on the missing parameters. In Figure 1.9, Payan et al. [8] organized several significant helicopter events into safety metrics and prioritized them based on their severity. Once the results are obtained from either exceedance analysis or event detection, statistical analysis is a useful tool to uncover trends and patterns.

In addition to the HFDM, the HUMS program focuses on monitoring the deterioration of mechanical parts inside helicopters, and it is primarily implemented to prevent system component failures. Typical parameters being monitored in the HUMS program are those from the engine, transmission, and rotor systems. These acquired sensor data are used to predict the failure times of the parts inside each system. With all information gathered

from the HUMS, an intelligent maintenance schedule can be established and, as a result, increase the airworthiness of the vehicle.

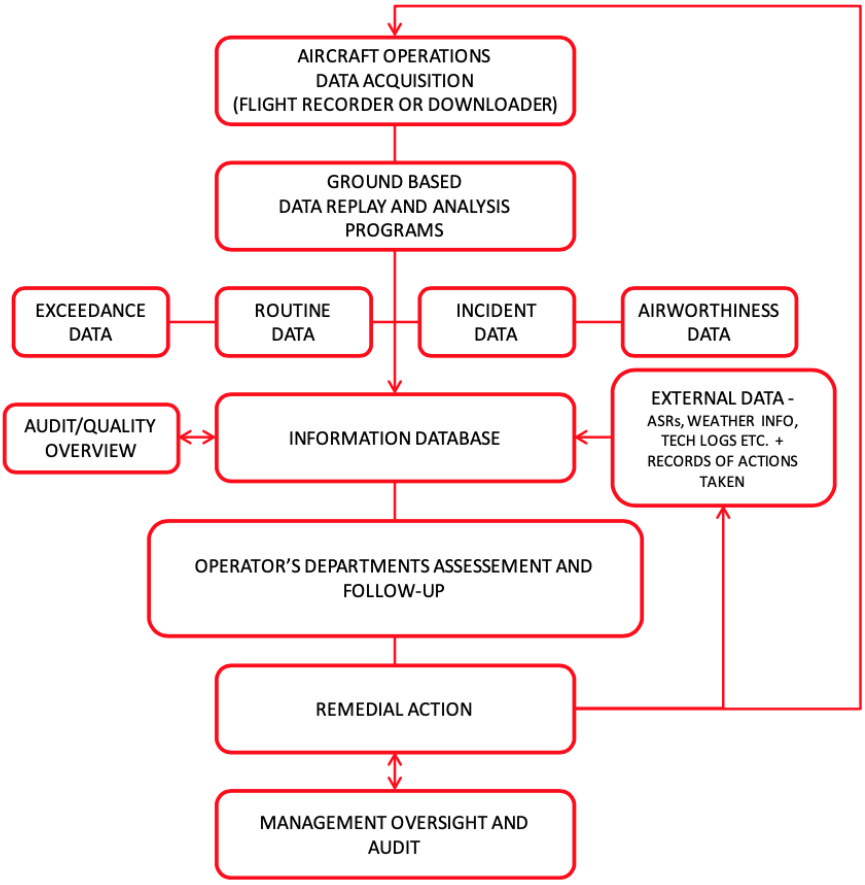


Figure 1.8: Process flow of a typical flight data monitoring (FDM) program [7]

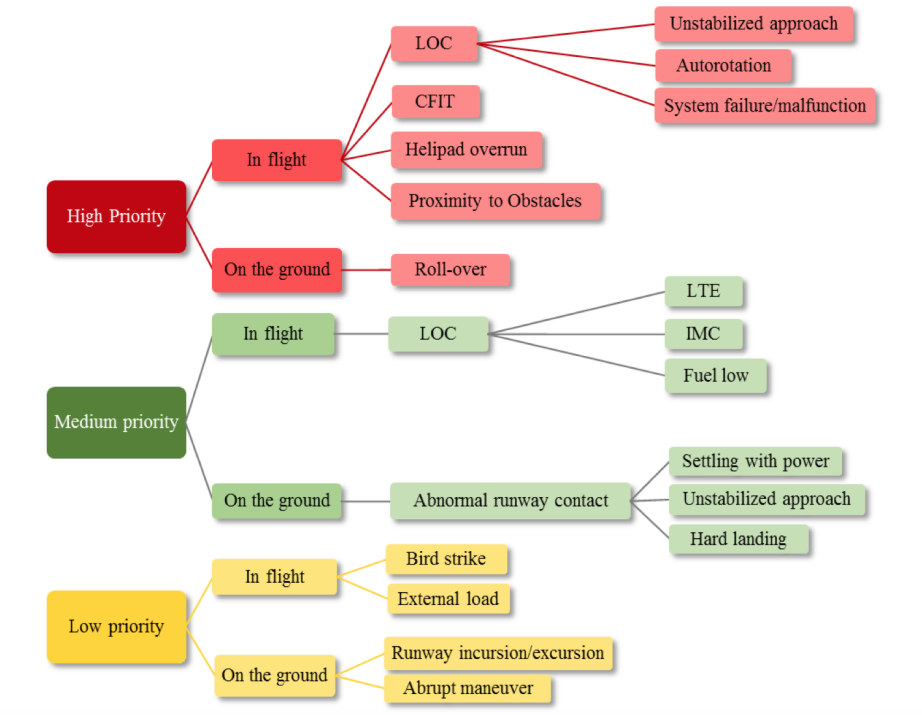


Figure 1.9: Categorization and prioritization of safety metrics for helicopter operations [8]

1.2 Motivation

In the early years when helicopters were first introduced to the aerial operation, they had suffered from high accident rates due to a lack of knowledge on the rotary-wing vehicle. With the improvements in rotorcraft design and system reliability, along with the experiences gained from accidents, the accident rate continued to decline until a plateau was reached. As pointed out in [9], one of the significant factors that affect the trend of the accident rate is the pilot's action. Compared to fixed-wing flights in commercial airliners, helicopter flights had higher accident rates during the period investigated in [10]. Thus, more attention needs to be directed in this domain. Further, a more recent report from the U.S. Helicopter Safety Team (USHST) [11] indicates that, even though the number of helicopter accidents had dropped for the past few decades, it started to go in a reverse direction since 2015. In order to gain a fuller picture of what is happening in an incident or an accident, the use of flight data recorders onboard helicopters is consequential, and this

action was recommended by the National Transportation Safety Board (NTSB) [12]. With that being said, the amount of flight data records available for investigation would grow. Therefore, it is necessary to have a toolkit for analyzing the data and giving feedback to the relevant parties based on the results from the analyses.

To support the recommendation of using flight data recorders for risk mitigation, the Federal Aviation Administration (FAA) initiated an effort to build an information system for monitoring flight data records. This endeavor is a collaboration between multiple organizations, including operators, USHST, the Helicopter Association International (HAI), and universities. The anticipated end product would be similar to the Aviation Safety Information Analysis and Sharing (ASIAS) used in fixed-wing commercial aviation but adapted to fulfill rotorcraft needs. In this program, flight data records from participating operators are provided to HAI under a Memorandum of Understanding (MOU), and sensitive information is removed before placing these data into a database. The research teams can then analyze the data and prototype relevant metrics for the platform. The program described above is illustrated in Figure 1.10, and this partnership is envisioned to create some useful analytic tools for providing insights on the corresponding operation. Some earlier research achievements can be found in [13][14][15]. Flight data records from the same database will be used in this study to continue the effort. During the course of the study, it is observed that the detection of some safety-related events would rely on the knowledge of phases of flight. This dependency motivates us to first tackle the phases of flight identification, and to treat it as a prerequisite for other successive tasks such as knowledge discovery or anomaly detection.

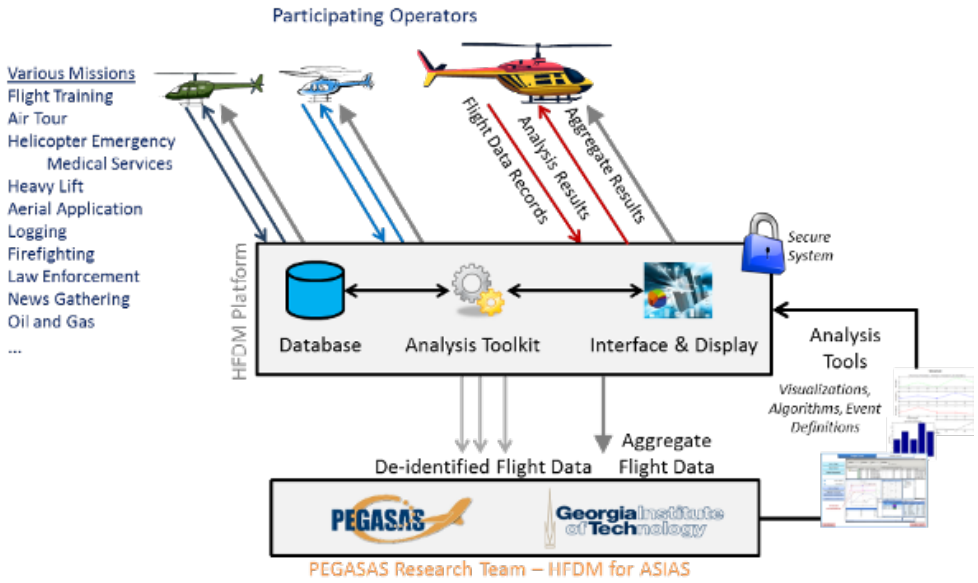


Figure 1.10: The concept of HFDM for rotorcraft ASIAS

Data science has become a popular area of study in the past decade. Many techniques in machine learning and data mining have been widely applied to various industries. Since flight data analysis is an essential element of the FDM, it is intuitive to leverage these methods to uncover the trend in the data. In the process of a typical machine learning task, the first step is to acquire the data that are sufficient for understanding the behavior of a system. Next, the techniques suitable for achieving the objective are selected, and the corresponding analyses are conducted. With the results summarized from the analyses, it is expected that the patterns of a system can be revealed. In general, these techniques can be categorized into two separate classes: (1) supervised learning and (2) unsupervised learning. The main difference between these two classes is that the former is suitable for classification and prediction tasks while the latter is more capable of pattern recognition. The selection of the methods within these two classes would primarily be determined based on the availability of the training data. It is anticipated that the safety of aerial operations can be enhanced by applying the appropriate techniques to detect abnormal patterns in the flight data, and providing constructive feedback to pilots or the management team. Numerous methods have already been implemented to analyze flight data records from

commercial airliners, and their efficacy is demonstrated in the relevant literature. Given the dissimilarity between the operations of helicopters in general aviation and fixed-wing flights in commercial aviation, some modifications may be required if we intend to leverage those methods to the rotorcraft domain. Some key aspects worth to be examined when considering applying data science methods to flight data records from rotorcraft operations are described as follows:

- Diversity: flight data records from commercial and fixed-wing general aviation usually follow a routine structure based on the corresponding mission profile. However, the flight profiles from rotorcraft operations may be more diverse due to various mission types ranging from search and rescue, air ambulance, power line cleaning, to even offshore operation. In Figure 1.11, an example of categorizing the flight data records into different hierarchical levels is provided. The heterogeneous nature of the helicopter flight data encourages the need to group similar flights before the analysis, and only with a proper grouping, a valid comparison can be guaranteed.

- Scalability: flight data recorders typically can record ten or more flight parameters, including flight parameters and even engine variables. With multiple vehicles operated daily, a large amount of data is required to be processed. The performance of the analytic tool should not degrade significantly as the sample size grows. Suppose the turnaround time for the analysis is considerably lengthy; in that case, the pilots or the management team may not get the feedback promptly, and it could lead to a higher risk in operation.

Aside from using post-flight analysis to enhance the safety of rotorcraft operations, knowing how to avoid hazardous events and react optimally upon encounters are also essential for accident mitigation. This approach is more proactive because it aims to guide pilots during the encounter of the event rather than afterward. The operational space can be thought of as the safety envelope of a hazardous event, and the process of exploring

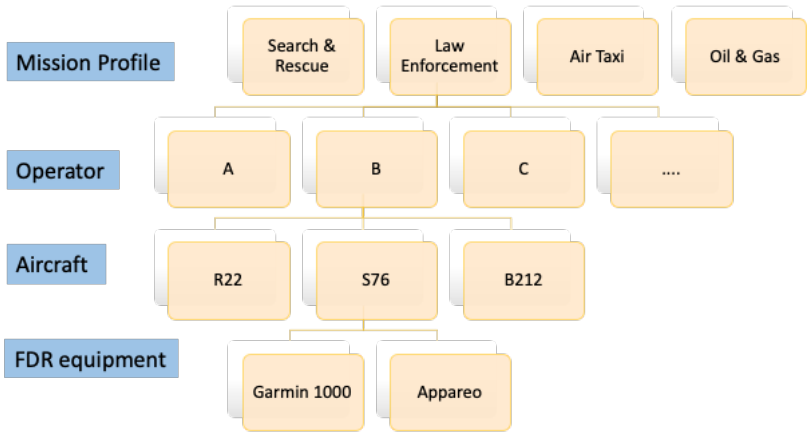


Figure 1.11: Different levels of categorization for flight data records

the space might be time-consuming based on various model complexity and optimization algorithms. Since the surrogate modeling with the design of experiments is typically used in the design space exploration, this approach may be suitable for operational space exploration. Two characteristics of a hazardous event considered in this study are rareness and time criticality. These events are rarely observed in a typical FDM program, and pilots could have less experience dealing with them. A flight simulator is a valuable tool to explore the unknown regions of the flight envelope, and it can also be used to train pilots. Another feature of a hazardous event is time criticality, which means that remedial action must be taken as soon as the events start. The delayed implementation of required recovery would be detrimental to the operation and increase the accident risk. Because the autorotation meets these two characteristics and also ranks the top two occurrence categories in the accidents from the compendium report [4], it is adopted as the use case for our exploratory analysis.

Autorotation is a special type of maneuver specific to rotorcraft, and pilots would apply this maneuver when helicopters experience engine failure. It is called autorotation because the rotor is self-rotated to keep the vehicle afloat. During the event, no power is supplied to the main rotor to produce the lift for sustaining the weight of the vehicle. In the autorotation, the main rotor is primarily driven by the airflow from beneath, and this phenomenon

is similar to a windmill effect. The key to a successful autorotation depends on how pilots manage the remaining energy in the rotor system. In the beginning, pilots would experience a sudden drop in altitude. If pilots try to pull the collective, which is the control for increasing the lift, it will stall the rotor blades and quickly lose energy. Therefore, the FAA helicopter flying handbook [16] instructs pilots to lower the collective initially for preventing rotor stall. Suppose the vehicle descends too fast without sufficient horizontal speed; in that case, the rotor will enter into vortex ring state (VRS), a condition where the rotation energy is mostly used for producing vortices rather than generating the lift. It is a challenging task for pilots to execute the maneuver appropriately under time constraints. Thus, finding the optimal control trajectories that properly manage the remaining energy to achieve a safe landing is important and knowing how to avoid the event in the operational space is also consequential. The height-velocity diagram (H-V diagram, usually called "deadman's curve" in the community) is a tool for pilots to understand the dangerous zone of particular combinations of height above ground and horizontal velocity. A typical H-V diagram is shown in Figure 1.12. Pilots need to avoid the red regions for the entire flight to ensure a higher margin of successful autorotation upon engine failure. In the past, the H-V diagram was constructed through real test flights. It could be dangerous for those cases closer to the real boundary. With the availability of a simulator and an appropriate surrogate model, it is anticipated that the operational space can be better understood under different scenarios.

1.3 Relevant Studies

In this section, we will first review various techniques used for detecting anomalies in flight data records from commercial fixed-wing aviation. To the best of our knowledge, there are few data mining methods applied to flight data records from general aviation, especially rotary-wing, potentially due to their heterogeneous nature and limited availability. It would be interesting to learn how these existing approaches can be transformed or leveraged to the rotorcraft domain. Since flight data records are in the forms of multivariate time series

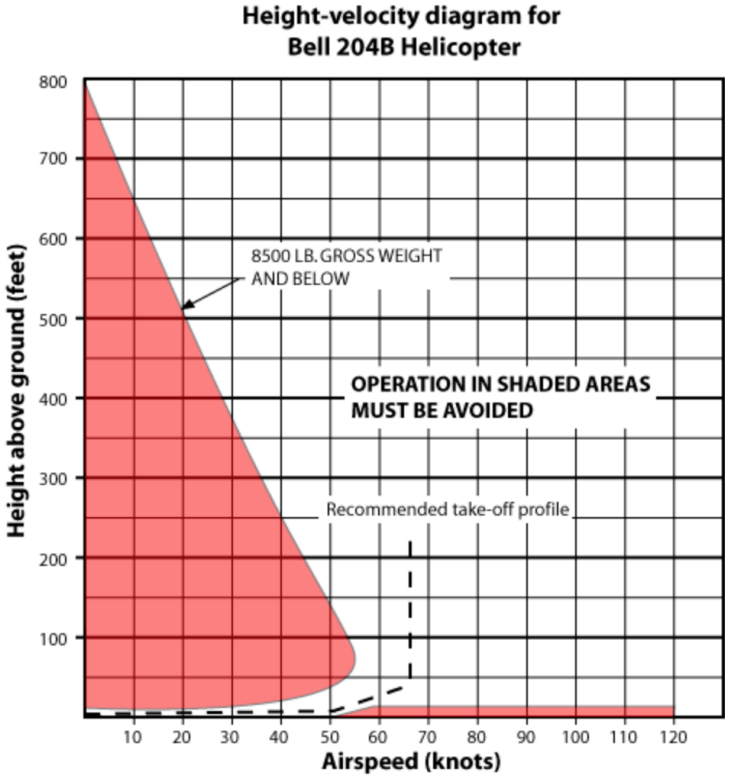


Figure 1.12: Typical height-velocity diagram [17]

and trajectory data, it is worth reviewing some recent data mining advancements on these data types. More specifically, some studies relevant to time series clustering and trajectory-based clustering were surveyed. The methods found in the literature might be useful for detecting the spatial-temporal patterns and anomalies in the data.

For the safety envelope exploration of the autorotation maneuver, the studies relevant to computing the optimal control were reviewed to support the work on the exploratory analysis. Since surrogate modeling will be looked into to predict the safety boundary and the corresponding controls, several types of research, including the design of experiments and surrogate models for predicting functional responses, will also be covered.

1.3.1 Data mining application in flight data records

Amidan et al. [18] developed a methodology called *morning report* for finding anomalous flights in commercial fixed-wing aviation. This methodology aims at relieving subject

matter experts from screening large amounts of flight data "at night" by instead presenting them potential anomalous flights the next "morning." First, the flight data records are truncated into homogeneous segments, which correspond to phases of flight. Then, for a specific flight phase, the multivariate time series is transformed into mathematical signatures, which are essentially statistical properties of a quadratic polynomial fit of the original time series. In the analysis, extracted mathematical signatures were clustered using the k-mean algorithm. Each flight in its cluster was assigned an atypicality score, and it is a metric created based on principal component analysis. For dealing with multiple flight phases in a standard flight sequence, a more general metric called *global atypicality score* was proposed. It is the combination of the p-value of the atypicality score and the cluster membership score. All the flights considered in the study were ranked based on this metric, and SMEs can use this information to identify anomalous flights.

Iverson [19] used a statistical-oriented and data-driven approach named Inductive Monitoring System (IMS) to find anomalies in flight data records. IMS is a supervised learning approach where training data are required to construct the knowledge database. This training set contains only the data from nominal operations, and the anomalous data were excluded. The knowledge database consists of high-dimensional clusters formed by the training data. Once it is constructed, the user can query the database for a newly observed data point to see if it belongs to one of the nominal clusters. If a data point is close to the boundary of a nominal cluster or far away from the cluster center, then it is flagged as an anomalous point. This approach is similar to process monitoring in which the knowledge database is the control chart rather than several groups of clusters. Temperature monitoring sensor data in the wings of the STS-107 Columbia Space Shuttle were used to demonstrate the capability of the IMS. The author claimed that the abnormal behavior in temperature reading could be detected 3 minutes after the foam impact. This information could have given crew members more time to find a countermeasure for alleviating the damage. A few improvements on top of this framework mentioned in the paper are (1) dimensional reduc-

tion of the parameter space and (2) to consider other clustering methods, such as distance or density-based ones, for constructing the knowledge database.

Budalakoti et al. [20] proposed a method named *sequenceMiner* which aims at detecting anomalies within discrete flight data records. A discrete flight parameter can be the signal from a control switch which can be toggled on and off in the cockpit. This parameter is different from a continuous flight parameter that typically represents the response of a dynamical system. In this study, discrete flight parameters or sequences were clustered using CLARA (Clustering LARge Applications) with the normalized Longest Common Subsequence (nLCS) as the distance measure. The sequences far away from the medoid of a cluster are declared anomalous sequences. The overall methodology is built upon a probabilistic Bayesian network, and it outperforms the Hidden Markov Model (HMM) in a test using synthetic data. From the analysis of a real dataset, *sequenceMiner* is capable of detecting 13 suspect sequences out of 2200 while the SMEs confirm that 5 out of the 13 detected sequences are truly anomalies.

Das et al. [21] developed a method called Multiple Kernel Anomaly Detection (MKAD), which seeks to tackle both continuous and discrete types of flight data records. In Multiple Kernel Learning (MKL), a generic kernel, fundamentally a weighted sum of individual kernel functions, is used to determine outliers based on the corresponding decision boundary. This idea is similar to ensemble learning since each kernel has its prediction functionality. In MKAD, one kernel is dedicated to continuous data while the other is for discrete data. In the study, the continuous data are discretized using Symbolic Aggregate approXimation (SAX), which can alleviate the impact of noise. Through this transformation, both types of data are in discrete format, and the corresponding kernels are chosen to be the normalized longest common subsequences. The MKAD was applied to 2500 flights in a descending phase within the same airport in the test. The number of flight parameters recorded is 160, and this set is shrunk to 39 based on subject matter experts' screening. Five hundred flights were used to train the model, and as a result, MKAD can identify 227 anomalous flights.

Within the identified anomalous flights, 114 flights can only be detected if both continuous and discrete data types are present in the analysis.

Chu et al. [22] proposed a method based on the idea of multivariate statistical process control (SPC), and they intended to use this method for detecting anomalies for flights in the cruise phase. In earlier studies, input and output signals from the system were studied separately, and no dynamical model was involved in the analysis. In this study, a surrogate of the dynamical system was built based on regression in which historical input/output pairs of data were used to train the model. To enhance the goodness-of-fit of the model, both affine and quadratic regressors were considered in the modeling. Once the surrogate model is trained, newly observed input/output data can be inserted into the model, and the residuals computed from the regression are served as the monitoring signals. The metric used in the control chart for signal monitoring is the Hotelling T^2 statistics. In the test, a dataset of simulated flights with fault injection was analyzed, and it is reported that the method is capable of finding 80% of the anomalous flights.

Das et al. [23] developed a framework called DMKD (Data Mining and Knowledge Discovery), and it is a multistep approach. The process of DMKD is shown in Figure 1.13, and it includes five steps: (1) data pre-processing (2) feature extraction (3) anomaly detection (4) consulting with subject matter experts (5) reporting. In feature extraction, the continuous data are first transformed into discrete features using Symbolic Dynamic Filtering (SDF) and then added to the discrete data. The combined set of features is fed into an outlier detection method called *iOrca* to find the anomalies in the data.

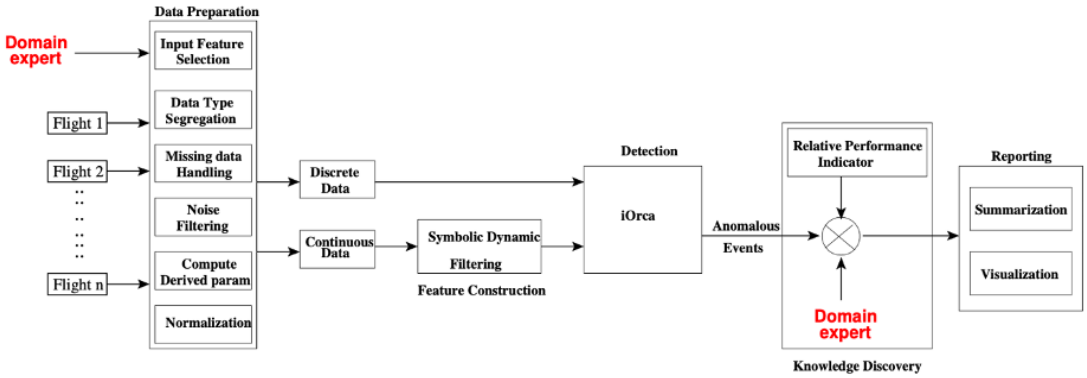


Figure 1.13: Data Mining and Knowledge Discovery (DMKD) Framework [23]

iOrca [24] is a distance-based anomaly detection method based on the idea of nearest neighbors and it is an improved version of *Orca* developed by Bay et al. [25]. In the nearest neighbor algorithm, there are two tuning parameters, k and t, where k stands for the number of nearest neighbors and t is the number of points with the largest values of the outlier metric. Each point inside the dataset has its own k-nearest neighbors, and its outlierness can be measured by the mean distance from itself to the k-nearest neighbors. A point with a higher value in the outlier metric is relatively far away from its neighboring points than a point with a lower value in the outlier metric. Once the outlier metric for each point is found, all data points are ordered based on their values in the outlier metric, and the top t points with the largest values in the outlier metric are flagged as outliers. This algorithm is easy to implement. However, its computation time scales with the square of the size of the dataset. *Orca* improved the computation time for the nearest neighbor algorithm by introducing the concepts of blocking and pruning. The dataset is truncated into several subsets by blocking, and the nearest neighbor algorithm is applied sequentially on each block. By pruning the points that are not likely to be outliers, the number of distance calculations is reduced. With these two modifications, *Orca* can bring the computational time down to a linear relationship with the size of the dataset rather than a quadratic relationship. *iOrca* goes one step further by indexing the points in the dataset so the list of outliers can be updated faster compared to *Orca*.

The Symbolic Dynamic Filtering (SDF) is a method that enables the transformation of continuous data into a discrete format. The continuous data can be realized as a trajectory in p-dimensional space, and the feature space can be partitioned into a finite number of discrete cells that are mutually exclusive and exhaustive. As the trajectory intersects with a cell, the original data is replaced by the symbol associated with this cell. By using this mapping, the continuous data are transformed into sequences of symbols, and as a result, they are in discrete format. These sequences are then used to construct a D-Markov machine. The corresponding transitional matrix and stationary state probability are viable candidates to represent the new features for the anomaly detection step. The concept of SDF can be found in Figure 1.14, which is illustrated by Rao et al. [26].

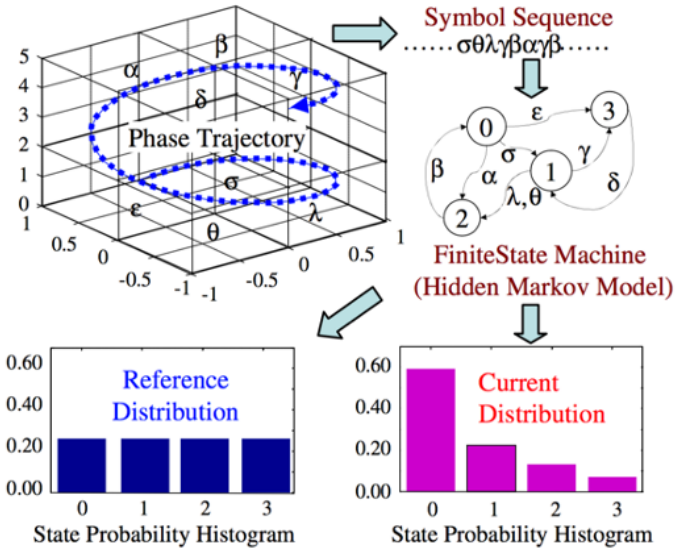


Figure 1.14: Symbolic Dynamic Filtering [26]

To verify the DMKD [23], a dataset that contains 25,519 flight segments from the phase of landing with the same destination was tested. Further, the authors investigated a reduced set of flight parameters within each flight segment, including both continuous and discrete variables. They discovered that using the SDF for feature extraction can detect more anomalous flights compared to not using the SDF. Also, they found that the anomalous events detected using the DMKD are associated with those defined in the FOQA program.

Li et al. [27] came up with a data-driven method called ClusterAD (Cluster-based Anomaly Detection) to compare multiple flights within selected flight phases. The analysis was conducted for flight segments under the same flight phases because a normal event in one phase might be abnormal in the other phase. Unlike the supervised learning approach, ClusterAD is designed to find the abnormal behaviors without knowing the nominals in advance. The dominant structure observed from most of the flights should define the nominal pattern. There are three significant steps in the ClusterAD: (1) data transformation (2) dimensional reduction and (3) clustering analysis. In data transformation, multivariate time-series data are stacked into a gigantic vector to ensure that all variables' effects are included in the analysis. In order to perform the analysis in a relatively low-dimensional space, principal component analysis (PCA) is applied to capture the essential features in the step of dimensional reduction. For the clustering analysis, the algorithm of Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is selected, and it has the benefit of no need for specifying the number of clusters in advance. In the test of ClusterAD, 365 B777 flights were investigated, and the selected flight phases are the takeoff and the approach. Among all approach segments being analyzed in this study, two anomalies were detected: the deviation from the nominal approach speed and the abnormal flap position. Two events were identified for the takeoff segments, including the reduced power takeoff and change of takeoff power. In [28], the performance of ClusterAD was compared with MKAD and exceedance detection(ED), which is the current standard used by airliners. Results showed that ClusterAD and MKAD outperform ED in identifying significant operational anomalies. Moreover, ClusterAD worked well for continuous flight data, while the MKAD had more detection power when dealing with discrete sequence data. In [29], Li et al. developed a variant of the ClusterAD called ClusterAD-DataSample, which uses a Gaussian Mixture Model (GMM) instead of a clustering algorithm for anomaly detection. The revised framework is capable of finding local anomalies in specific timestamps. As a result, the detected anomalies are more interpretable than those that all timestamps are

tagged as anomalies.

In the general aviation domain, Puranik et al. [30] used energy metrics along with raw flight data records as the features to perform the task of anomaly detection on flights in the approach phase. In this method, the Euclidean distance is selected as the similarity measure for feature comparison, and DBSCAN is applied to detect outliers. Several anomaly scores were implemented, including the local outlier factor and the average KNN distance. If the flights are associated with high anomaly scores, they are tagged as outliers in the group. The proposed framework was demonstrated to detect anomalies in simulated and real flight data records.

Deshmukh et al. [31] developed a temporal logic-based learning method called TempAD to detect anomalies for flights in terminal airspace operations. First, the flights with similar patterns in horizontal trajectories are placed in the same group using DBSCAN clustering. Once the flights are separated into distinct trajectory patterns, a discrete structure search and a continuous parameter search are performed to construct the upper / lower bounds for a flight parameter. These linear boundaries acquired from the analysis can then be used to detect anomalies retrospectively and monitor the flight parameters in real-time. This method is different from the exceedance analysis because the thresholds are learned through the data rather than pre-defined based on experiences.

Jarry et al. [32] applied Functional Principal Component Analysis (FPCA) to detect anomalies for commercial fixed-wing flights in the approach phase. In this method, a sliding window is designed to move through the energy profile to extract the first k principal component coefficients. Then, to calculate the outlier score for a particular segment, the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) is applied to the extracted coefficients. In the study, some atypical events, such as the deviation in ground speed, were detected, and these events do have an impact on the operation.

Memarzadeh et al. [33] used a convolutional variational autoencoder (CVAE), which is one of the deep generative models, to detect anomalous commercial flights in the takeoff

phase. This method falls into the realm of unsupervised learning; thus, no labeled dataset is required for the training. The structure of CVAE consists of one encoder followed by a decoder, and flight data records are compressed and recovered in the process. The flights with reconstruction errors higher than the threshold are regarded as anomalies. The method was first tested using a benchmark dataset with labels, and then its capability of detecting anomalous flights was demonstrated on the flights in a FOQA dataset.

Two studies are noteworthy in reviewing data mining techniques on flight data records. Gavrilovski et al. [13] conducted an extensive review on some prior work in the field and also pointed out some challenges and opportunities for leveraging the methods developed in the general aviation domain. More recently, Basora et al. [34] categorized the methodologies used for anomaly detection in the context of flight data monitoring and predictive maintenance. They also indicated a growing trend of using neural networks and deep learning to detect anomalies in large datasets.

1.3.2 Data mining on time series and trajectory data

Flight data records consist of various flight parameters, and they are typically in the form of multivariate time series and trajectory data. Therefore, it is useful to review some studies relevant to pattern mining of these data types. In [35], Liao did a comprehensive survey on the techniques used in time series clustering. He first reviewed some standard methods used to cluster static data and then explained different routes for transitioning to time series clustering. Potential approaches for clustering time series are summarized in Figure 1.15, which is extracted from [35]. Unless the raw time series are used directly, additional steps like feature extraction or time series modeling are required to retrieve representative information before performing clustering analysis. A few recent studies in time series clustering were organized into the categories mentioned above and can serve as references for future research.

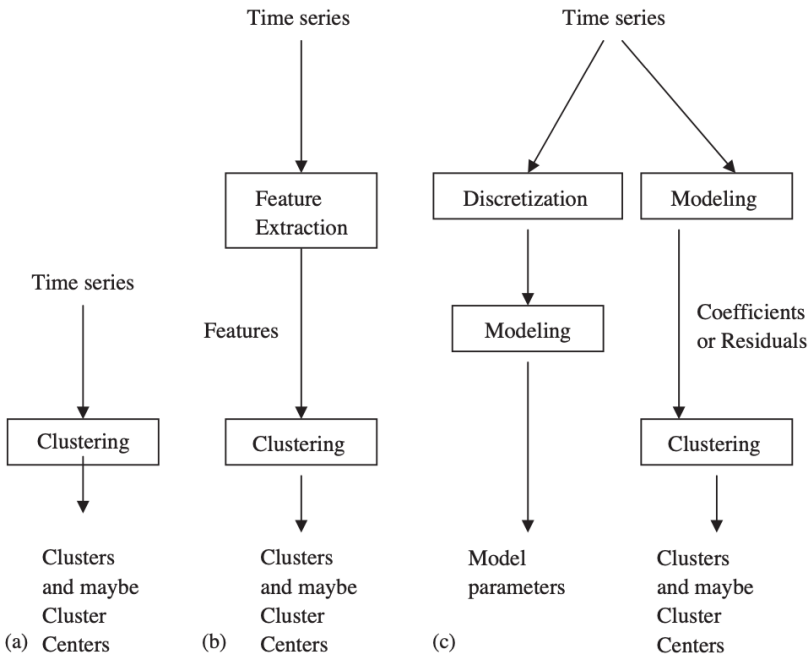


Figure 1.15: Different approaches for time series clustering [35]

Aghaborzorgi et al. [36] surveyed a large number of studies on the application of time series clustering in a variety of fields. In this review, four key aspects of time series clustering were discussed. Further, as pointed out by the authors, two mainstream approaches were taken by the majority of the studies: (1) transform high-dimensional time series data into a low-dimensional representation and then apply traditional clustering algorithms (2) apply similarity measure for calculating distances between time series without extracting features out of raw time series. Blazquez et al. [37] provided a review on studies of detecting anomalies in time series. In this research, a taxonomy of anomaly detection in time series was proposed, and the techniques used for detecting different types of outliers were discussed and summarized.

For finding patterns in trajectory data, Zhang et al. [38] performed trajectory clustering using different similarity measures on a labeled outdoor surveillance dataset. They found that more straightforward measures such as the Euclidean distance can outperform other complex metrics in terms of detection efficiency. Zheng [39] provided a comprehensive

overview of the process for trajectory data mining. A roadmap, which includes several essential steps such as data preprocessing, data management, data mining, and data transformation, was given as guidance for researchers. Different similarity measures for trajectories were introduced with the application of various clustering strategies. In the section on anomaly detection, some studies related to finding outliers in road traffic patterns were mentioned. Su et al. [40] compared the effectiveness and efficiency of several similarity measures for comparing trajectories. To account for the issue of poor data quality in a real scenario, trajectory transformations, such as point shift, shape stretching, and noise addition, were applied in the experimental test. The measures capable of handling most of the transformations were identified, and it was shown that their effectiveness comes at the cost of efficiency.

1.3.3 Optimal control in autorotation

The optimal control problem (OCP) is an optimization problem attempting to find the preferable control profiles in functional space for a dynamical system. The computed control profiles optimize one or more designed objective functions and satisfy the constraints. The OCP has its roots in the calculus of variation, and the first OCP, the Bachistocrhone problem (path with the shortest time) posed by Bernoulli in 1696, is the first one investigated by the mathematician at the time. However, the optimal control theory was not fully-fledged until the modern era (1960), and this field of study is primarily based on the works by Pontryagin (Minimum principle) and Bellman (Hamiltonian-Jacobi-Bellman equation). An example of an OCP could be moving a cart with an inverted pendulum from one location to another in minimum time, or finding the trajectory for a rocket from the earth to the moon with minimum fuel.

Over the past 30 years, researchers have shown great interest in tackling the autorotation problem. A substantial effort has gone into finding the safety boundary and optimal path for a safe landing. These studies explored different aspects of the problem, including the type

of vehicle, model complexity, operating condition, and the strategy for computing optimal control. Some highlights of these studies noted below were presented in chronological order, and to better characterize these studies, they were grouped by the methodology or the research team.

Indirect Method

Johnson [41] was the first to investigate the controls required to land the rotorcraft under power loss. A two-dimensional point-mass model was used to study the autorotation. In the study, hovering autorotation was the primary focus because the helicopter is expected to perform better in cases with initial forward velocity. Here, an indirect method that calculates the controls through minimizing the Hamiltonian was implemented. As a result, the study found that the optimal trajectory for hovering autorotation is a vertical descent. The effects of different entry altitudes and lock numbers on state variables, such as touchdown velocity, were also explored.

Lee [42] used a model similar to [41] and extended the work by including cases with a forward velocity at the time of engine failure. It is also postulated that the area of the restriction zone in an H-V diagram discovered from the actual flight tests can be reduced by leveraging the nonlinear optimal control technique. The performance index was chosen as the square of the terminal speed, and the optimization is subject to equality and inequality constraints. The equality constraints are the equations of motion, while the inequality constraints are imposed to prevent the rotor stall and the excessive sink rate. The optimal control problem was solved using a two-step iterative process called Sequential Gradient Restoration Algorithm (SGRA). The objective function and constraint errors are minimized in the gradient and restoration phases separately. In the case of hovering autorotation, the author noted that the technique used in the flight tests was different from the controls computed by the algorithm. In the flight tests, pilots would attempt to achieve zero vertical speed a few feet above the ground before touchdown, and it usually took a longer flight

time for the entire maneuver compared to the trajectory calculated by the algorithm. In the cases of the autorotation with a forward velocity, the effects of different entry speeds and altitudes were studied and compared. The maximum entry height was set as 400 feet, and a metric called most critical entry height (MCEH) was proposed. If the initial altitude deviated from the MCEH, the available rotational energy would be better retained. The author also pointed out potential directions for future research, including adding wind shear effect and developing pilot-guided systems for autorotation.

Okuno et al. [43] aimed to predict the H-V diagram using optimal control theory. The helicopter was modeled as a two-dimensional rigid body with an additional degree of freedom in pitching angle compared with the point-mass model. The increased induced velocity in the vortex ring state and the effect of blade stall are included in its aerodynamic module. In this study, the author tried to accomplish two tasks: (1) to use an optimization process to predict the H-V curve, and (2) to find the optimal control procedure for landing the aircraft given an initial condition. The first task was achieved by minimizing the region covered by representative points (high hover, low hover, and knee points). In contrast, the second task was accomplished by minimizing the sum of square of the touchdown speed. The solution method adopted here is the SGRA, which is the same as [42]. The predicted shape of the H-V diagram is similar to the one derived from the actual flight tests. The effect of different model fidelity on the shape of the H-V diagram was also examined. One exciting aspect mentioned in the study is that the predicted knee points would be located at different spots in the H-V space for different initial flight path angles. When the engine fails during a climb phase, the restriction zone becomes larger (the knee point is located more to the right) than when the engine fails during a descent phase.

Before Zhao et al. [44], the work on optimal autorotation focused solely on single-engine helicopters. This research extended prior studies by investigating the condition of one engine inoperative (OEI) for a multiengine helicopter using a two-dimensional point-mass model of UH-60A. Four different scenarios were investigated, including rejected take-

off, continued landing, continued takeoff, and balked landing. The objective function for the rejected takeoff and the continued landing is to minimize the horizontal distance between the takeoff and the landing. For the cases of continued takeoff and balked landing, an additional objective function, which serves to minimize the altitude drop, was added to the formulation. The SGRA was applied to find the solution of optimal trajectory for state and control variables. It was found that more power is required for vehicles with a higher gross weight for landing or achieving a steady climb rate in OEI conditions.

Direct Method

In [45], Carlson et al. worked on analytically predicting the H-V diagram using optimal control theory but shifted the focus from a helicopter to a tiltrotor vehicle (XV-15). The vehicle was modeled as a two-dimensional rigid body with only longitudinal motion. Aerodynamic forces and moments on different components are calculated through aerodynamic coefficients, which are prepared as a table based on the curve-fitting results on the experimental data. The tiltrotor vehicle has two operational modes that can be modified by controlling the nacelle angle. Two kinds of autorotation maneuvers were mentioned in the study. For low-altitude autorotation, the vehicle would tend to come straight down and cushion before touchdown. For high-altitude autorotation, the vehicle initially would build up some forward speed and then flare before the end of the maneuver. The solution method used in this study is direct collocation, and it transforms the optimal control problem into a nonlinear programming problem by stacking all state and control variables in discrete-time. In comparison with the indirect method used in [41] - [44], it does not need to construct the necessary and sufficient conditions for the optimal control problem. Besides, it is reported from Betts [46] that the basic indirect method has a robustness issue on the initial guess of the adjoint variables. From the results of predicting the H-V diagram in the condition of one engine inoperative (OEI), the higher the gross weight of the vehicle, the larger the restricted zone. For predicting the H-V diagram in cases of all engine inoperative (AEI),

the author noted a convergence issue of high hover points for the vehicle with higher gross weights. Sensitivity analysis was also conducted to investigate the effects of various parameters, such as the number of nodes in the discretization process, delay in pilot control actions, and modification of model parameters.

In [47], Bachelder et al. developed a pilot-guided system for the autorotation training program using an optimal control approach. With the guidance given to pilots, the program aimed to reduce their workloads and increase the probability of a safe landing. A two-dimensional point-mass model similar to [42] was adopted with the addition of a first-order response equation for the engine. The objective function used in the formulation contains two parts: (1) a terminal cost for minimizing the touchdown speed and (2) a running cost used to regulate rotor rpm and the tilt of tip-path-plane (TPP). The direct collocation method was applied to find optimal trajectories, and the resulting parameter optimization problem was solved using a commercially available software program. In the paper, the framework was first validated by comparing its results with the flight test data of the OH-58A. Then the H-V diagrams of the OH-58A and the SH-60B were predicted and the calculated touchdown speeds were compared with the maximum speeds documented in the flight manual. The authors found that H-V restriction zones computed by the optimal control technique are significantly smaller for both of these helicopters than the results derived from the actual flight tests. This finding supported the hypothesis posed by [42].

In [48], Jhemi et al. demonstrated the usage of the direct collocation method for finding the optimal control for autorotation. The paper briefly explained the process of transforming the optimal control problem into a parameter optimization problem. It also addressed the challenge of computing controls in real-time given higher model fidelity and the existence of external disturbances. Two cases were investigated in the study, including (1) flight path optimization of helicopters under power loss and (2) the prediction of the H-V diagram for tiltrotor vehicles.

In [49], Carlson et al. used an analytical method in conjunction with flight test data to

predict the H-V diagram for upgraded helicopters. Again, a two-dimensional longitudinal point-mass model with the first-order response of the engines was selected. This generic helicopter model was verified by comparing the power required at different trimmed conditions with the values found in the flight tests. , The direct collocation was selected to compute the optimal control trajectories due to its larger convergence radius and straightforward problem formulation. The H-V diagram was predicted by minimizing an objective function, which is constructed based on the distance between a reference point inside the envelope and the points at the edge. As a result, the shape of the H-V diagram can be parametrized by the power ratio, which is the ratio between the power available at the OEI and the power required for the hovering out of ground effect (HOGE). For different operating conditions, pilots can use the power ratio value to find the corresponding H-V diagram.

Differential Dynamic Programming (DDP)

Pieter et al. [50] developed an autonomous controller for a remote-controlled (RC) helicopter under the condition of power failure. This paper had demonstrated the first controller being implemented in an actual autorotative flight, although it was not dedicated for a full-scale, human-crewed vehicle. Instead of using a physics-based model for controller development, several test flight data were acquired to construct a three-dimensional rigid-body model through system identification. The model contains 13 state variables, which include translational positions and rates, rotational positions and rates, and rotor speed. For the control variables, lateral cyclic and rudder were added to address this three-dimensional model's roll and yaw motions. There are three primary stages for autorotation, namely the glide, the flare, and the touchdown. The expected vehicle behavior in an autorotative glide is to maintain a steady descent to better retain the remaining rotational energy. The vehicle would end in a level position with no forward velocity for an ideal autorotative touchdown. The flare is the most demanding task out of the three, and appropriate control for pitching

the nose is essential to avoid the tail boom strike and a hard landing. In the study, the target trajectories of the flare were provided by expert demonstration, and the Differential Dynamic Programming (DDP), a non-linear version of Linear Quadratic Regulator (LQR), was used to track these trajectories. The resulting controller was able to land an RC helicopter 25 times, and it was shown that the simulated results were close to the ones from the flight tests. Although it demonstrated several successfully autorotative landings for an RC helicopter, the validity of the controller for other strenuous initial conditions has not yet been verified.

Reinforcement Learning

In [51][52], Lee et al. used Reinforcement Learning (RL) to tackle the autorotation landing problem. In the methodology, a reward function is designed to drive the agent, which is a helicopter in this case, to a designated goal with optimal / sub-optimal state and control trajectories. In the process of learning, the agent would adapt itself through several iterations of trial and error. In this analysis, a model-free reinforcement learning approach called Q-learning was selected, and the radial basis function (RBF) was used to approximate the value function to circumvent the issue of the curse of dimensionality in Dynamic Programming (DP). A two-dimensional point-mass model similar to [42] was used, and the controls were modified to second derivatives of the thrust coefficient and the TPP tilt for shrinking the size of the action space. The reward function was designed to award the agent if the final touchdown speed is within the acceptable limit. The agent would be penalized in cases like hard landings and descending too fast during the maneuver. Two test cases, including the hovering autorotation and the autorotation with little initial forward velocity, were used to demonstrate the methodology. The strategy for the hovering autorotation found in this study was to increase the thrust coefficient gradually without changing the TPP tilt. For the cases with little initial forward velocity, the TPP was required to tilt backward for slowing the vehicle at the touchdown. The policies learned in both cases were valid for an altitude

lower than 100 feet above ground level with low airspeed. If the initial conditions are outside of the specified region, the agent might need to be re-trained. Further, there was no guarantee that the computed policies would work under environmental disturbances.

Studies from Penn State University

In [53], Tierney et al. tried to find a set of feasible initial conditions that would guarantee the helicopter to land successfully on a specific spot at ground level. Instead of investigating the whole process from the start of the engine failure to the touchdown, the study focused on the second half of the autorotation process, which was the portion starting from a trimmed steady descent stage to the final landing. By choosing the appropriate steady descent conditions and the timing for initiating the flare maneuver, pilots could land the vehicle successfully without power. The author called these feasible initial conditions the "backward reachable set." In the study, a two-dimensional point-mass model of a utility helicopter, which is similar to [47], was adopted and the ground effect was ignored. The objective function contains both the running cost and the terminal cost. Any exceedance from the path constraints would be penalized as running cost, and the deviation from the specified final touchdown condition would contribute to the terminal cost. The result of the safe landing set was projected onto the space of the horizontal speed/height to construct the V-h diagram. This V-h diagram is different from traditional H-V diagrams because it depicts the safety region of trimmed autorotative descent rather than the safe entry conditions. In sum, given the existence of a trajectory following algorithm which can guide the vehicle from the start of engine failure to the safe landing set, this methodology can assist in landing the vehicle onto a specific spot successfully.

Yomchinda et al. [54] continued the effort of [53] and tried to find the optimal path of the vehicle from the entry to the safe landing set using a modified Dubin's curve. Dubin's curve is a trajectory planning method for finding the shortest path between two points for a wheeled vehicle with a constant velocity/turn rate. The allowable controls of the wheeled

vehicle are the right turn (R), the stay straight (S), and the left turn (L). It was found that the possible optimal control sequences are LSR, LSL, RSL, RSR, RLR, and LRL, which are all three-segment sequences. To account for vertical motion of the vehicle, a modified Dubin's curve approach was implemented. For this investigation, a two-dimensional point-mass model was extended to three-dimensional by introducing additional variables in both state and control spaces. The whole process was assumed to be in a quasi-steady state, which means that the horizontal acceleration, the rotor speed, and the bank angle are constant in each of the segments. Also, the side-slip velocity was assumed to be zero for simplification. The trajectory planning problem was solved using parameter optimization by minimizing the deviation from the specified final altitude and rotor speed. The author demonstrated several successful cases with different initial and desired end conditions. However, it was noted in the paper that good initial guesses are essential for the optimization process. Without the proper selection of initial guesses, the solution might have been trapped in the local optimum or taken a significant amount of time to converge. Also, to make the flight autonomous, a controller must follow the computed trajectories.

Grande et al. [55][56] increased the complexity of the problem in [53] by introducing the wind shear effect. The wind shear profile was defined as a logarithmic profile in altitude, and only the horizontal wind conditions were considered. This effect was evaluated via different combinations of wind strength and wind direction. The wind velocity was added to the force balance and kinematic equations in helicopter flight dynamics to account for the wind shear effect. The study examined two different types of helicopters, the OH-58A, and an electric-powered helicopter. For the OH-58A helicopter, a safe landing set can be found in all test cases except those with medium and strong tailwind conditions. The helicopter would gain excessive horizontal speed in tailwind conditions, making it harder to slow down at the touchdown point. Further, the authors pointed out that a light headwind condition was beneficial for autorotation landing since it allowed a wider range of initial horizontal speeds. A safe landing set only exists in light and medium headwind

conditions for the electric-powered helicopter due to a higher susceptibility to wind shear for small-size helicopters.

Other miscellaneous studies

There are other studies relevant to the optimal control for autorotation, and various techniques were applied to the problem. Without elaborating on the details, we will only mention some critical points of each reference. Dalamagkidis et al. [57] proposed to use nonlinear model predictive control (NMPC) with a recurrent neural network (RNN) for calculating the optimal trajectory in real-time. However, the controller cannot be successfully demonstrated in real-time due to the model discrepancy and environmental disturbances. With improved model accuracy, objective function, and better training data, this framework can potentially work as an online controller. Bibik et al. [58] developed an autorotation model with higher degrees of freedom and solved for the optimal control using a formulation with a linear system and a quadratic cost. It aimed at deploying the controller for an autonomous autorotative flight, and different scenarios like AEI and OEI were considered in the study. The capability of this method was demonstrated in several test cases. However, because the code is computationally intensive, the method cannot be applied in real time. Sunberg et al. [59] developed an expert system based on control laws to achieve successful landing in autorotation. In the process, the autorotation maneuver was separated into different sub-phases, and for each sub-phase, a PID controller was implemented to reach the designed objective. The simulation results from two different helicopters demonstrated the capability of the algorithm. Still, it might be a tedious process for tuning the parameters for the controllers. Eberle et al. [60] developed a tau-based (time-to-contact) automatic autorotation controller to generate flare trajectory and to track a landing point. For the flare trajectory generation, the computed trajectory was compared with several trajectories from pilot-in-the-loop simulations, and it was found that these two are similar strategically. Two cases with different distances from the flare entry to the touchdown were

tested for the landing point tracking, and the algorithm can successfully land the vehicle in both situations.

1.3.4 Design of experiments and surrogates for functional data

The safety envelope of the autorotation can be thought of as a restriction zone in the H-V diagram. The curves which dictate the restricting areas can vary based on different conditions. The surrogate modeling with the design of experiments can be potentially applied to find these restriction zones in the operational space. Design of experiments (DOE) is a tool for efficiently exploring the response of a system. It has been widely used in a variety of areas, such as system design and industrial / manufacturing processes. For example, to use this framework for designing a new vehicle, a mathematical representation that captures the physical system's behavior needs to be formulated. A corresponding computer model is then required to be developed for experimentation. If this computer code is computationally expensive, a surrogate model, also referred to as an emulator or a meta-model in literature, is required. It can be used for a variety of purposes, such as prediction, optimization, and variable screening. In [61], Queipo et al. described the structure and key elements of a surrogate-based analysis. A few selections of DOEs and surrogate models were introduced in the study. In addition to predicting the system's response, other applications of surrogates, such as sensitivity analysis and optimization, were discussed. The methodology was substantiated in a test case of designing a liquid-rocket injector, and it is more effective to incorporate surrogate modeling into the design. Joseph et al. [62][63] proposed a new type of space-filling design called Maximum Projection Design (MaxPro). In the study, various space-filling designs suitable for computer experiments were discussed and compared. It is demonstrated that the MaxPro design has better space-filling property in subspace projection than the widely used maximin Latin hypercube design (MmLHD). A machining simulation was provided to illustrate the capability of this new design, and it was shown that the surrogate model well represents the simulator.

Since we would like to predict the optimal control for a hazardous event and the controls are functional responses, some studies relevant to using surrogate models for predicting functional responses were reviewed. Hung et al. [64] developed a Gibbs sampling-based Expectation-Maximization (EM) algorithm to tackle function responses in a non-regular grid. A use case for predicting the residual stress of a machining process was demonstrated in the paper. Gul et al. [65] proposed an approach to quantify the uncertainty of functional responses for a milling process. The study also considered the cases with multiple types of input data, including quantitative and qualitative variables. Mak et al. [66] developed an efficient emulator to predict the time evolution of the flow field pattern for a rocket injector. The cross-section velocity field of an injector is in the form of image data, and it is a step beyond the cases of predicting functional responses. It was stated in the paper that the emulator could efficiently capture the spatio-temporal pattern of a highly accurate but time-consuming CFD model.

1.3.5 Observations from literature

From the studies pertaining to using data mining on flight data records, it is observed that the majority applied the approaches to flight data coming from fixed-wing commercial aviation rather than from general aviation, not to mention from the rotary-wing domain. In general, rotary-wing flights exhibit more diverse patterns than their fixed-wing counterparts due to the capability to hover and the ability to satisfy the need for a variety of operations. As such, it is more challenging to identify the norm and the abnormality of flight data records without well-defined definitions of homogenous segments for characterizing helicopter flights. In addition, the analyses found in the studies were conducted on flight segments within the same flight phase rather than the entire flight data records. Among the flight phases investigated in prior work, the most analyzed flight phases are the approach and the takeoff. Due to differences in operations between rotorcraft and fixed-wing vehicles, phases of flight appropriate for fixed-wing aircraft may not be directly applicable to

rotorcraft. They may need to be modified or add extra flight phases to the set. It is also expected that the transitions between flight phases are more frequently observed in certain rotorcraft operations in comparison with larger fixed-wing aircraft in commercial aviation.

Concerning the practice of anomaly detection in the industry, exceedance analysis is the most widely-used approach. However, it can only detect the predefined events, and it might be challenging to have a universal threshold for an event given various helicopter types, mission profiles, and operators. In this situation, data mining could potentially be applied to identify patterns in the flight data without specifying thresholds on flight parameters. Eventually, the pattern recognition activity results may be translated into anomaly detection after subject matter experts (SME) provide some inputs into what is an anomaly and what is not. Further, it is time-consuming to build a fully-fledged HFDM program for the operators due to the development of safety metrics. With the availability of flight data records at a fleet level over a certain period of time, some insights and knowledge from the data can be extracted. The information acquired from the analysis can possibly contribute to defining the thresholds of the events. Thus, the process of building an HFDM program for detecting anomalies can potentially be shortened through the application of data mining methods.

In general, data mining methods can be separated into two categories, and the literature reviewed fall into these two sets:

1. unsupervised learning [18][20][23][27][29][30][31][32][33]

2. supervised learning [19][21][22]

The main difference between these two categories is the requirement of a training dataset for model parameter estimation. For supervised learning approaches, sufficient data with labels should be provided for training purposes. If rare events and corner cases are included in the training set, the algorithm can typically produce high accuracy results. However, it is difficult to obtain a large labeled dataset with high quality in real life, and it is probably why the majority of the research would adopt unsupervised learning approaches. When dealing

with an unsupervised learning method, synthetic or simulated data are used for validation before deploying the algorithm to the actual dataset.

For the exploratory analysis in autorotation, most of the studies mentioned above used a two-dimensional point-mass model for its simplicity in the formulation. Only a few studies developed a more sophisticated three-dimensional model for the investigation. Although the point-mass model has its limitation, such as not being capable of showing the flare and turn maneuver in autorotation, it is still quite useful in terms of the safety envelope exploration. Aside from the model complexity, the prediction of the safety envelope for autorotation in previous studies was made primarily based on the curve fitting of a few representative points in the H-V space. Suppose more test points are available in the entire region of the H-V space. In that case, we can better understand the effect of a variety of initial conditions on the performance metric. Moreover, depending on the selection of model complexity and optimal control algorithm, solving the problem of trajectory optimization may be computationally intensive. Last but not least, the safety envelope is constructed on the space spanned by two variables. Different combinations of variables other than the height and the horizontal velocity, such as the vehicle weight, should be considered in operational space exploration.

1.4 Research objectives

In the previous section, various strategies for improving the safety of flight operations from the literature are reviewed. A few observations from the review are summarized, and the associated gaps are also addressed. In this section, the overarching research objectives are stated to fill the gaps and the scope of the study is determined. This study will mainly focus on two different approaches: retrospective and exploratory analyses. The retrospective analysis is a data-driven, learn from the past approach. It is intended to identify the norms and potential anomalies in operation from analyzing sufficient flight data. Some relevant machine learning techniques would be investigated in the analysis for recognizing patterns

in the data. Subject matter experts (SME) could potentially review the investigation results, and then the consensus from SME's discussion will be forwarded to the management team/pilots for improvement. This feedback system is similar to the practice in FDM programs, and it is helpful to keep the rotorcraft operation in a healthy condition.

The first research objective regarding the retrospective analysis is stated as follows:

> **O1**: To improve flight safety specific to rotorcraft operation through retrospectively discovering potential anomalies in flight data records.

For the exploratory analysis, the overarching goal is to build a viable framework to explore the safety envelope for a hazardous event efficiently. Therefore, the second research objective is stated as follows:

> **O2**: To develop an efficient methodology to explore the safety envelope and acquire the recovery trajectory in a hazardous event.

The exploratory analysis is a model-based, proactive approach. It aims to use a flight simulator to explore hazardous events that may or may not be observed from flight data records in an FDM program. Using a simulator for this type of exploration reduces the risk compared to an actual flight test and can significantly expand the number of scenarios in the investigation. There are two different tasks that we plan to pursue in the exploratory analysis: (1) safety envelope exploration and (2) optimal control prediction in the operational space.

In the following few chapters, several research questions will be formulated separately to address individual research objectives. For each research question, potential methods for solving the questions are hypothesized. To substantiate the ideas proposed for tackling the problem, we have designed several experiments to test these hypotheses and to consolidate those ideas further. Different topics, including phases of flight identification, anomaly detection on flight data records, and exploratory analysis for the autorotation, will

be described in chapters 2, 3, and 4 correspondingly.

CHAPTER 2

PHASES OF FLIGHT IDENTIFICATION

The first research objective for the study is to find the potential anomalies in flight data records for rotorcraft operations using data mining methods. To initiate the task, we first emphasize the need for having an algorithm for detecting flight phases of rotorcraft operations. Several flight phases definitions from different sources are reviewed, and a baseline definition is established based on the findings. Further, potential techniques useful for the identification task are presented, and they are compared based on the classification results of a labeled dataset and the proposed evaluation criteria. The experiment described in this chapter aims to find the best or combination of methods to detect flight phases for rotorcraft operations.

2.1 Problem formulation

From the review of the literature as mentioned earlier, it is intuitive to ask the following research question (RQ1) as the first step to move toward our first research objective:

> **RQ1**: Without reinventing the wheels, can we leverage the anomaly detection methods used in commercial airliners to general aviation, and more specifically, to the rotorcraft society?

The hypothesis (H1) corresponds to this research question is stated as follows:

H1: The anomaly detection tasks in commercial airliners are implemented for flight segments within the same phase of flight. Given that the homogeneous segments can be retrieved for helicopter flights in general aviation, those methods can be potentially applied.

Without the enabler that could identify the flight phase information from flight data records, it is challenging to make a valid comparison among the flights and apply some practical methods used in commercial aviation for detecting anomalies. Therefore, the algorithm of phases of flight identification is treated as the cornerstone for the retrospective analysis. Here, the RQ1 and the H1 are stated to complete the overall logic flow, and thus, no experiment is conducted. However, the H1 leads to our second research question, and it is stated as follows:

RQ2: What methods or techniques can be helpful to find the homogeneous segments, i.e., phases of flight, for the rotorcraft operation?

The hypothesis to address the second research question is constructed as

H2: The phases of flight for the rotorcraft operation can be potentially retrieved through a filtering approach, regression-based classifiers, and the supervised learning methods used in the task of classification.

The experiment will be performed for testing the H2 is described in the following:

E2: To find the appropriate method for the task of phases of flight identification, it is necessary to compare and contrast the methods suggested in the H2 using the classification accuracy of a labeled dataset and some proposed evaluation criteria.

2.2 Approach for detecting flight phases for rotorcraft operations

2.2.1 Phases of flight definitions

Survey on available phases of flight definitions in the literature

To start off the task, we first surveyed the available definitions for phases of flight in the literature. These definitions were gathered from different sources, and some of them are dedicated to rotorcraft operations [16][67] while others are relevant to flights in commercial airliners [68][69] or flights in general aviation [70]. The results of the survey are summarized in Table 2.1 and some opinions from subject matter experts along with potential flight parameters involved for defining the flight phases are included as well for reference.

Table 2.1: Available definitions of flight phases in the literature

	FAA [16]	ICAO [67]	ECCAIRS [68]	FAR AIM [69]	GAPOF [70]	Subject matter experts	Parameter (s) Mentioned
Standing		Prior to taxi or after arrival while the aircraft is stationary			*Groundspeed* = 0 kt Indicated *airspeed* = 0 kt		Groundspeed Indicated Airspeed (IAS)
Taxi	*Altitude*: Surface taxi: 0 AGL Hover taxi: <25 ft AGL Air taxi: <100 ft AGL	Prior to takeoff or after landing	*Groundspeed*: < 37 km/h (20 kt)	*Altitude*: Hover taxi: <25 ft AGL Air taxi: <100 ft AGL *Groundspeed*: >20 kt *Altitude*: > 3 rotor diameters	*Groundspeed* < 25 kt Indicated *airspeed* = 0 kt		Groundspeed IAS Altitude AGL
Takeoff	Adjacent phases may be hover and standing	From the application of takeoff power to 35 feet above runway elevation	From the application of takeoff power to an *altitude* of 50 feet above runway elevation		Engine *RPM* > 2500 RPM *Altitude*: < 35 ft AGL		Engine RPM Altitude AGL
Climb		From the end of takeoff to the first prescribed power reduction, or reaching 1000 feet AGL	From the end of takeoff to the first prescribed power reduction, or reaching an *altitude* of 1500 feet AGL		Change in *altitude* > 200 fpm	Gaining altitude at a *rate* larger than 200 fpm over 20 sec	Vertical speed Duration
Cruise	Constant *altitude*, heading with a desired airspeed	Intermediate phase of "en route"	Same as ICAO [67]		Change in *altitude* <= ±200 fpm	Altitude change within a *rate* of 200 fpm over 20 sec	Vertical speed Duration Altitude AGL
Descent	Loss of *altitude*	Last portion of "en route"	Same as ICAO [67]		Change in *altitude* < -200 fpm	Losing altitude at a *rate* larger than 200 fpm over 20 sec	Vertical speed Duration Altitude AGL
Approach	Descent profile: Normal approach: 7-12 degree @ 300-500 ft AGL Steep approach: 13-15 degree Shallow approach: 3-5 degree	From the end of "en route" to the beginning of the landing flare	Same as ICAO [67]			Stable approach should have a *rate of descent* less than 1000 fpm	Altitude AGL Approach angle Vertical speed
Landing		From the beginning of the landing flare until aircraft exits the landing runway	Same as ICAO [67]				
Hover	Constant *altitude* and constant *heading* with nearly no motion over a reference point			*Altitude*: > 3 rotor diameters			Altitude AGL Groundspeed Vertical speed
Turn			Parameters related: change in *heading* and *bank angle*				Heading Bank angle

After retrieving these definitions on flight phases from the literature, some observations

are noted here concerning the applicability of these thresholds on helicopter flights. First, no consensus was found regarding using constraints on flight parameters for defining flight phases. Thus, it might take some efforts of trial-and-error to determine the thresholds for defining the flight phases for rotorcraft operations. Further, some definitions such as the ones in [68] [69] are explicitly designed for larger aircraft for the mission of passenger transportation. Without modification, they may not be immediately applicable to helicopter flights.

For flights in commercial airliners, the patterns of flight phase sequences are reproducible because the flights usually follow standard routes in the operations. On the rotorcraft side, depending on the types of missions, a helicopter flight may go through numerous climb, cruise, descent phases. Due to the ability to hover, additional flight phases like hover in-ground/out-of-ground effects, hover lift/descent are required to be included in the phases of flight identification for the rotorcraft operation.

In [16], one can find the most relevant definitions of flight phases for helicopters. However, the descriptions are presented in the format of the pilot's actions for achieving the maneuvers rather than threshold values on flight parameters. In fact, the sequences of control inputs may be helpful in retrieving flight phases because they are closely related to the pilot's intent for steering the vehicle. However, most flight data recorders do not contain these signals in the flight parameters. As a result, the approach of using the control inputs for detecting flight phases for helicopters is excluded in the study, and they are not shown in the last column of Table 2.1 for conciseness.

In Table 2.1, some flight phases are delineated using qualitative descriptions rather than quantitative thresholds on flight parameters. A transcription process is required to have these definitions used in an algorithm for detecting flight phases. Again, the process might involve a few trial-and-error iterations, and the thresholds chosen may be subjective. Moreover, the definitions of three flight phases, namely, the descent, the approach, and the landing, may overlap given they all have the characteristic of loss of altitude in common.

To differentiate these flight phases, additional logic may need to be provided. For example, the loss of altitude can only be an approach if the vehicle is heading toward its destination, not the other points of interest.

Interpretation on the definitions of flight phases from the literature

To build a filtering logic for detecting flight phases for rotorcraft operations, we combined the definitions in Table 2.1 and transcribed them into quantitative constraints on flight parameters. The manifolds selected in the flight parameter space for defining the flight phases should fully cover each data point in flight data records. Otherwise, some timestamps in the flight data would be labeled as unidentified. Further, these manifolds should not intersect with each other for guaranteeing no timestamp would be simultaneously classified as multiple phases. In [71], Kelly et al. investigated the concept of using a fuzzy set for identifying specific flight segments. Although a single timestamp might correspond to several potential phases, a membership function that describes the probability of being in each candidate phase can be used to determine the actual flight phase for that specific timestamp. This study motivates us to develop a method in which several probable flight phases can be assigned to a single timestamp in flight data records.

In the last column of Table 2.1, flight parameters that are useful for defining the flight phases are listed. It is observed that the most commonly used parameters are the horizontal velocity (either ground speed or airspeed), the vertical velocity (climb or descent rate), and the altitude. Therefore, a filtering approach which is based on these three flight parameters will be first attempted, and it is noted that the flight phases are not necessarily confined by only three flight parameters. The benefit of this choice is twofold: it is easier to visualize the manifolds for defining flight phases in a three-dimensional space, and these three parameters can be easily retrieved across all kinds of flight data recorders. For completeness, we do not exclude the possibility of adding more flight parameters for more accurate identification in the future study. The following paragraphs explain the selection of threshold

values for defining flight phases for rotorcraft operations. If the quantitative values can be extracted from the aforementioned literature, they will be used directly. Otherwise, the qualitative descriptions will be transcribed based on our own interpretation.

Standing phase

When the helicopter is in a standing phase, the vehicle is on the ground with no horizontal or vertical velocity. Thus, the ground speed and the altitude are used to capture this flight phase. Conceptually, we should set these two flight parameters at exact zero, but to account for noisy signals, they are allowed to vary in the ranges of [0, 0.6) knots and [0, 2) feet, respectively. The vertical velocity is not involved in the definition because the standing phase is sufficiently characterized by confining the vehicle on the ground level without a significant forward motion.

Taxi phases

In [16], the taxi phases are clearly defined, and three kinds of taxis are mentioned in the handbook. These taxi maneuvers are differentiated through altitude ranges, and for low, medium, and high altitude ranges, the corresponding taxi maneuvers are surface taxi, hover taxi, and air taxi. For the horizontal velocity, it is specified that the speed should not exceed the one for a brisk walk; thus, it is set to vary between 0.6 to 18 knots.

Hover phase

When the helicopter is in a hovering phase, the vehicle is situated at a higher altitude without significant movement in both horizontal and vertical directions. It can be thought of as a standing phase at high altitude, and here it is described using three flight parameters. The horizontal speed constraint is the same as the one in the standing phase, while the altitude should be higher than the one used for a standing condition. The limitation on the vertical speed is set as [-90, 90] feet per minute to account for the gust or noisy signals. The hover phase can be further separated into three sub-categories: (1) hover in-ground effect (HIGE) (2) hover out-of-ground effect (HOGE) and (3) high altitude hover. The first two can be identified through a threshold value in height (two times the rotor diameter), while

the remaining one is recognized when the vehicle is at an even higher altitude.

Climb / Cruise or level / Descent phases

For these three flight phases, the vertical speed is an important parameter for capturing the corresponding flight segments. In a climb phase, the climb rate should continuously exceed a threshold value while in a descent phase, the descent rate should always stay higher than the same threshold value. Anything not a climb or a descent would be assigned to a cruise or a high altitude hover phase. In this study, a threshold value of 90 feet per minute on vertical speed is selected based on preliminary testing on a flight school's data. An altitude constraint is placed on these three flight phases to distinguish them from other low altitude flight phases, such as the standing and the taxi phases.

Takeoff / Landing or approach phases

The takeoff and the landing phases are treated as the transitional flight phases between the low-altitude and the high-altitude flight phases. The low-altitude flight phases consist of standing and different taxis and hovers. The high-altitude flight phases include the climb, the cruise or the level, the descent, and the high altitude hover. The takeoff can be realized as a flight phase when the vehicle is coming from a low-altitude region to a high-altitude region, and the landing can be defined in a reverse manner. Given the flight phases in low and high-altitude areas are identified, we can find the takeoff and the landing by labeling the interim periods. For example, the last portion of a low-altitude flight segment may be relabeled in conjunction with the first few seconds of the following high-altitude flight segment as a possible definition for the takeoff. In fact, a takeoff identification algorithm is proposed in the following subsection, and it is developed based on the aforementioned flight phases. For the approach detection, Robinson et al. [72] developed an algorithm to identify different types of approaches such as Visual Flight Rules (VFR) approach and Instrument Flight Rules (IFR) approach for helicopters. It can be incorporated into our primary phases of flight for completeness.

In sum, the threshold values used to define each of the flight phases mentioned above are

provided in Table 2.2. It is our baseline definitions of flight phases for rotorcraft operations, and a survey dedicated to verifying these definitions will be distributed to subject matter experts for feedback.

Table 2.2: Baseline definitions of flight phases for rotorcraft operations

Variable ---------- ***Phase***	**Vertical Speed [ft/min]**	**Ground Speed [kt]**	**Altitude [ft]**
Standing		< 0.6	[0,2)
Surface Taxi		[0.6,18]	[0,2)
Hover Taxi		[0.6,18]	[2,25)
Air Taxi		[0.6,18]	[25,100)
Hover	[-90, 90]	< 0.6	>2
Climb	> 90		≥100
Cruise	[-90, 90]	> 0.6	≥100
Descent	< -90		≥100

A survey to subject matter experts for verifying the baseline definitions

To verify the proposed threshold values in Table 2.2 for detecting flight phases, we sent a survey to some subject matter experts who are experienced professionals in the field. The qualifications of the subject matter experts invited to the study are listed in Table 2.3. From the survey results, the opinions from different experts do not go hand in hand with the definitions. For instance, one SME would treat the air taxi as a subset of the hover taxi, and the other would prefer to separate them into two flight phases. Nonetheless, we incorporated the feedbacks from SMEs and provided a set of updated definitions. The modified constraints are shown in Table 2.4. We will implement these constraints in a filtering approach for phases of flight identification, and this approach will be compared with other candidate methods.

2.2.2 Potential techniques for the identification

As mentioned in the previous subsection, we can retrieve the phases of flight for rotorcraft operations by applying filters on specific flight parameters. However, it is not the only

Table 2.3: Qualification of subject matter experts involved in the survey

Respondent	Occupation	Years in Occupation
1	Helicopter pilot	9
2	Helicopter pilot Safety officer; rotorcraft Accident investigator	20 5 3
3	Helicopter pilot Safety officer; rotorcraft Accident investigator	30 25 16

Table 2.4: Modified definitions of flight phases for rotorcraft operations based on the survey

Variable ---------- *Phase*	Vertical Speed [ft/min]	Groundspeed [kt]	Altitude [ft]
Standing	Close to 0	< 0.6	[0,2)
Surface Taxi	Close to 0	[0.6,20]	[0,2)
Hover Taxi	Close to 0	[0.6,30]	[2,25)
Air Taxi	Close to 0	[0.6,50]	[25,100)
Hover In Ground Effect	[-90,90]	≤ 0.6	< 2 blade diameters
Hover	[-90,90]	≤ 0.6	≥ 100
Hover Lift	> 90	≤ 0.6	< 100
Hover Descent	< -90	≤ 0.6	< 100
Climb	> 90		≥ 100
Cruise	[-90,90]	> 0.6	≥ 100
Descent	< -90		≥ 100

approach to acquire the information of phases of flight. Given a flight data record, the task of phases of flight identification is to predict the phase labels for all timestamps from a subset of flight parameters, and this task is similar to the classification in supervised learning. If the control inputs are available in the flight data, we can also use them to infer the flight phases. As a result of brainstorming, potential methods which can assist the task of identification are listed below:

- Filters constructed from thresholds on flight parameters
- Classification techniques used in supervised learning
- Regression-based classifiers
- Control inputs such as the collective, longitudinal / lateral cyclic, and the pedals

There are pros and cons associated with each of the approaches. For example, a filtering method is easy to implement once the corresponding threshold values are determined. However, the chosen values for constraints might need to be modified for a different type of operation. Sometimes, the flight parameter we would like to monitor may not even exist in the flight data records. In addition, to achieve high classification accuracy, the approach in supervised learning may need to have large amounts of data for training. In some instances, it might not have the luxury of accessing the labeled dataset. To eliminate the dependency on the training data, we can resort to an unsupervised learning approach. Still, it is not guaranteed that the clusters found from the analysis would be relevant to the phases of flight that we intend to detect.

We first selected a filtering method in the initial search for the algorithm because of its simplicity and feasibility. From the results of a preliminary test using the flight data from a flying school, two issues need to be addressed, and they are (1) unidentified timestamps and (2) flight phases with short durations. To address these issues, other approaches that may be potentially suitable for the task are investigated and introduced in the following subsections. These methods are grouped into two distinct categories: (1) the methods can be used directly without the training process (2) the models that required to be trained before deployment. An experiment conducted will be described in the following section to test their effectiveness. The investigation results can guide selecting a method or a combination of methods that can make a relatively accurate prediction on the phase label.

Flight phases in low and high-altitude regions

From observing flight phase definitions in the literature, it is feasible to divide the entire flight data record into flight segments into two distinct regions, namely, the low-altitude and the high-altitude regions. It is assumed that certain flight phases only exist in the low-altitude region while the remaining flight phases solely happen in the high-altitude area. The regions are distinguished by a threshold value in altitude, and typically it is set as 100

feet above ground level. The advantage of making this assumption is that there is no need to consider flight phases in the high-altitude region for data samples in the low-altitude region and vice versa. The shrinkage of the set of candidate flight phases for each timestamp could reduce computation time and improve the overall efficiency of the algorithm. The flight phases belonging to each of the regions are specified as follows:

- Flight phases in the low-altitude region: standing, surface taxi, hover taxi, air taxi, hover in ground effect (HIGE), hover out of ground effect (HOGE), hover lift, and hover descent
- Flight phases in the high-altitude region: climb, cruise, descent, and high altitude hover
- Transition flight phases between these two regions: takeoff and landing

Figure 2.1 illustrates the idea of placing individual flight phases into one of the two regions. We can determine the takeoff and the landing phases once all the timestamps in the flight data are assigned with flight phases either in low or high-altitude regions. In this study, we would primarily focus on the basic set of flight phases, and the takeoff and the landing phases are treated as add-on flight phases to the basic set. To substantiate the applicability of detecting the takeoff from the basic set of flight phases, an implementation of the takeoff identification will be explained later in the following subsection.

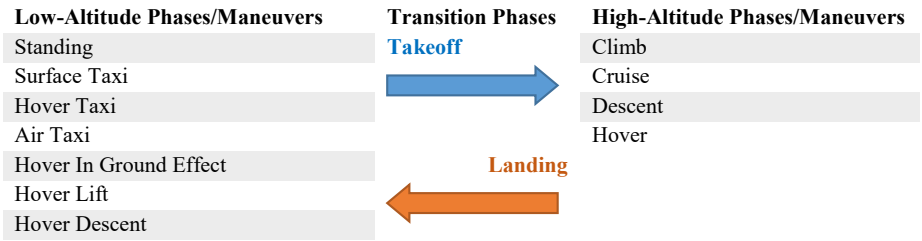


Figure 2.1: Flight phases in low and high-altitude regions along with transition phases

Piecewise linear regression

Piecewise linear regression (PLR) is an approach derived from the field of time series segmentation, and it is used to represent a profile or a curve in the time domain with a few small line segments. In [73], Keogh et al. introduced several algorithms, including top-down, bottom-up, sliding windows, and a mixed approach, to slice a time series into a handful of entities. Lovric et al. [74] also review on some popular methods used in the field. Through the PLR approach, we can reveal the general pattern of a time series without being affected by the noisy data points. The process of PLR starts from representing all data points in a growing window by linear regression, and the window will continue to expand until the regression line no longer fits the data points. Then, the process will repeat for the following data points, and it ends when all data points are consumed. A cartoon that illustrates the operation of the PLR can be found in Figure 2.2, and a pseudo-code is provided in Appendix A.1 for reference. In this study, this method is applied to the altitude signal for detecting the climb, the cruise, the descent, and the high altitude hover. If the data points are summarized by a line segment with an ascending trend, then all of the timestamps associated with these data points will be classified as the climb. The same logic is applied to detect the descent, and if a segment is not a climb or a descent, it is considered the cruise or the high altitude hover.

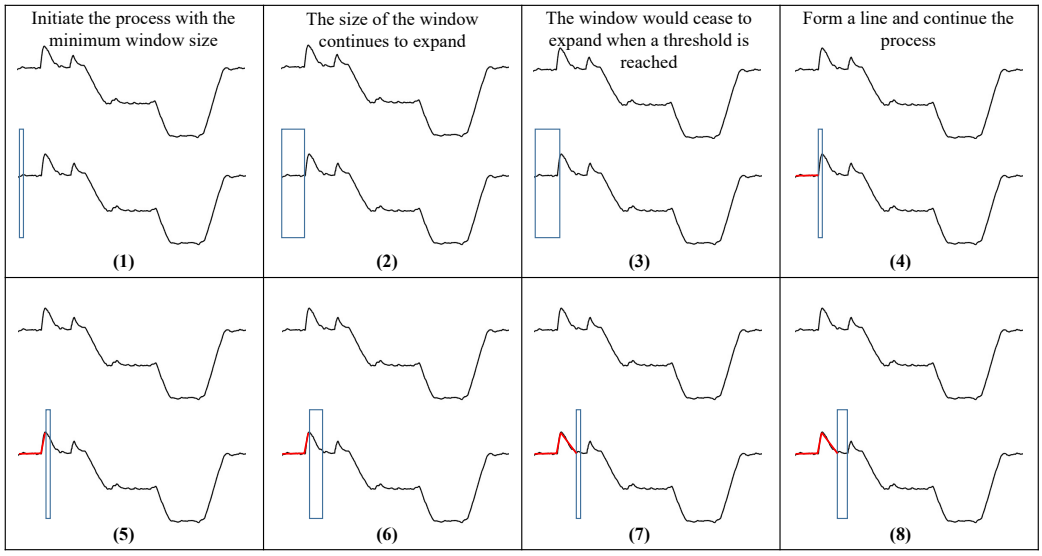


Figure 2.2: The process of the piecewise linear regression method (adapted from [74])

There are three parameters to tune for the PLR method: the initial size of the growing window, a threshold value on the goodness-of-fit for the regression line, and a slope threshold for distinguishing the climb, the cruise, and the descent. The first parameter would dictate the least amount of data points for forming a segment. When picking a larger value on this parameter along with noisy data points, the window would stop growing due to exceeding the threshold value on the total residual error. To pick an appropriate initial window size, we suggest starting from a smaller value and then monitoring the effect when larger values are applied. The second parameter directly determines how well a regression line represents the data points. For cases with noisy data, a larger threshold value would lower the impact of fluctuating signals. The last parameter is the slope threshold, and with a smaller value on this parameter, the cruise segments would be less likely to trigger than the cases with a larger slope threshold.

Sliding window regression classification

The sliding window regression classification (SWRC) is a regression-based classifier. It is motivated by a fuzzy set approach in which a timestamp could belong to multiple candidate

flight phases. Unlike the PLR method, the window size for SWRC is kept constant, and this window would slide through consecutive timestamps for performing regression until reaching the end of the data record. As such, most of the timestamps would be classified multiple times, and the prevailing label determines the flight phase for a specific timestamp. An illustration is shown in Figure 2.3. In the figure, the initial six realizations of the method with a window size of 3 are demonstrated, and the third point is classified as label L since the majority of the labels are associated with the third point are L. The pseudo-code of SWRC is presented in Appendix A.2.

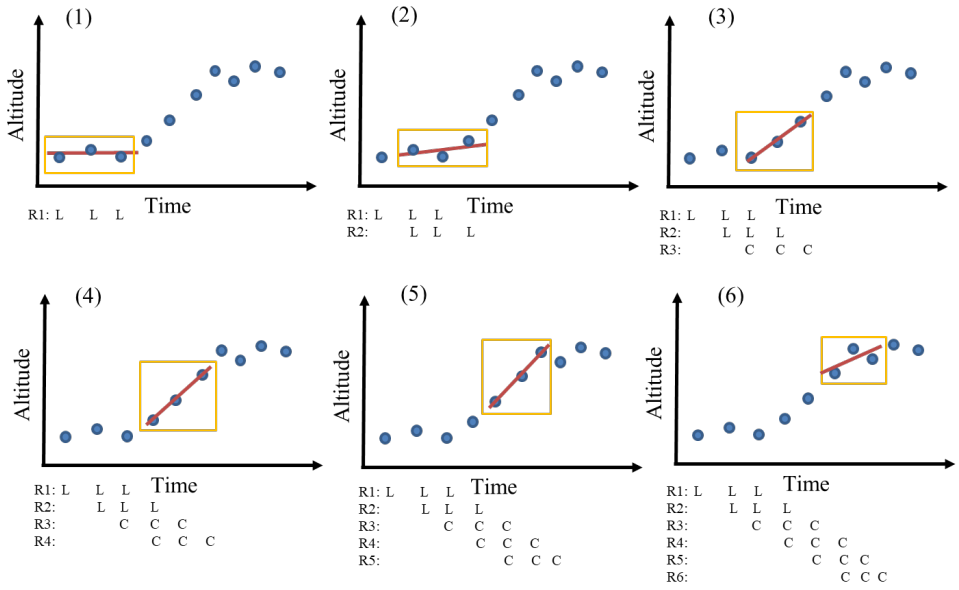


Figure 2.3: The process of the sliding window regression classification

Sequence smoother

In the preliminary test of applying filters to detect flight phases, it was found that some flight phases identified are within short durations, and these short duration phases would frequently flip from one to another. The same phenomenon is also observed from the test of the PLR method. To address the issue of flight phases in short duration, we developed an approach called sequence smoother to eliminate short duration segments. In the sequence smoother, a flight phase in a short time is combined with a neighboring flight phase in a long

duration. If there exist multiple flight phases in a short duration consecutively, they are first combined into a segment in a long duration, and then this new segment will be reclassified. Two scenarios are presented in Figure 2.4 to illustrate the concept of the sequence smoother, and the detailed implementation of this approach is documented in Appendix A.3.

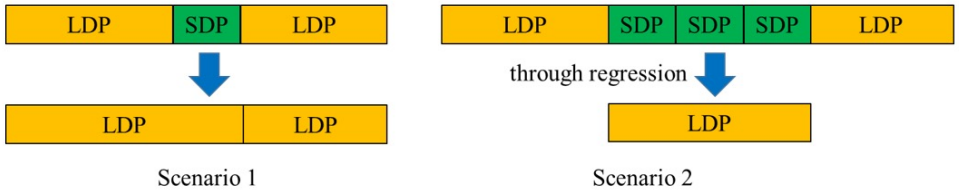


Figure 2.4: Using sequence smoother for dealing with short duration segments

Logistic regression

Since the phases of flight identification can be thought of as a classification problem, logistic regression, which is a method in supervised learning, can predict the flight phase labels. Compared with linear regression, logistic regression is used when the outputs are categorical labels rather than numerical values. In this model, the probability of predicting flight phase k given the observed data, sometimes called the posterior distribution, is taken the form of a sigmoid function. In the context of phases of flight identification, assuming there exist K flight phases to be detected, the posterior distribution of a specific phase k can be expressed in Equation 2.1 and 2.2:

$$P(\text{Phase} = k|X = \boldsymbol{x}) = \frac{\exp\left(\beta_{k0} + \beta_k^T \boldsymbol{x}\right)}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \beta_l^T \boldsymbol{x})}, \quad k = 1 \cdots K-1 \tag{2.1}$$

$$P(\text{Phase} = K|X = \boldsymbol{x}) = \frac{1}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \beta_l^T \boldsymbol{x})} \tag{2.2}$$

where β are the regression coefficients which can be derived from the maximum likelihood estimation and $\boldsymbol{x}$ is a vector formed by explanatory variables or predictors. The $\boldsymbol{x}$ can be any flight parameters that are relevant for predicting flight phases, such as the altitude,

the horizontal speed, and the vertical speed. By manipulating Equation 2.1 and 2.2, the following expression can be formed:

$$\log \frac{P(\text{Phase} = k|X = \boldsymbol{x})}{P(\text{Phase} = K|X = \boldsymbol{x})} = \beta_{k0} + \beta_k^T \boldsymbol{x}, \quad k = 1 \cdots K - 1 \tag{2.3}$$

The right-hand side looks exactly the same as linear regression, while the left-hand side can be interpreted as the probability of being in flight phase k over the likelihood of being in flight phase K. After all the regression coefficients β are learned in a training process, the $\boldsymbol{x}$ for each timestamp can be inserted to Equation 2.1 and 2.2 for prediction, and the phase with the highest posterior probability would be assigned to that specific timestamp.

Naive Bayes classifier

Another approach in supervised learning that can be used for detecting flight phases is the Naive Bayes classifier, and it is a method based on the Bayes' theorem. It is a probabilistic model, and the posterior distribution of being in flight phase k can be formulated as below:

$$P(\text{Phase} = k|X = \boldsymbol{x}) = \frac{P(X = \boldsymbol{x}|\text{Phase} = k)P(\text{Phase} = k)}{\sum_{l=1}^{K} P(X = \boldsymbol{x}|\text{Phase} = l)P(\text{Phase} = l)} \tag{2.4}$$

where k is one of the K phase labels and $\boldsymbol{x}$ is a vector comprised of all flight parameters involved for the prediction. Based on Bayes rule, the right-hand side of the Equation 2.4 is the product of the likelihood function and the prior distribution of being in flight phase k divided by a normalization constant. In this study, the likelihood function is assumed to follow a multivariate normal distribution, and it can be factorized into a product of several individual univariate normal distributions. Once the hyperparameters for the model are acquired from the training, the model can be deployed for the prediction. Given the observation $\boldsymbol{x}$, if the probability of being in flight phase k is higher than the probability of being in flight phases l where $k \neq l$, then the flight phase k would be the predicted result for the corresponding timestamp.

Linear and quadratic discriminant analyses

A variant of the Naive Bayes classifier, which also falls into the supervised learning, is the discriminant analysis. It has the same formulation as the Naive Bayes classifier but instead of modeling the likelihood function as fully factorized univariate normal distributions, it is represented as a multivariate normal distribution. In the linear discriminant analysis, it is assumed that the multivariate normal distribution from different classes would have the same covariance matrix. The decision boundary for the linear discriminant analysis can be computed using the following formula:

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k \tag{2.5}$$

where μ_k and Σ are the mean and covariance matrix of the multivariate normal distribution and π_k is the probability of observing flight phase k in the data. It is called a "linear classifier" since the decision boundary is a linear function with respect to the predictors $\boldsymbol{x}$. The phase label k is assigned if $\delta_k(\boldsymbol{x})$ is the highest value among all $\delta_l(\boldsymbol{x})$ where $l \neq k$. In the quadratic discriminant analysis, the assumption of equal covariance matrix is relaxed and the decision boundary for the quadratic discriminant analysis has the following form:

$$\delta_k(x) = -\frac{1}{2} x^T \Sigma_k^{-1} x + x^T \Sigma_k^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma_k^{-1} \mu_k - \frac{1}{2} \log |\Sigma_k| + \log \pi_k \tag{2.6}$$

where Σ_k is the covariance matrix for each of the flight phase k. The quadratic discriminant analysis is more complex than its linear counterpart since more hyperparameters have to be estimated. However, the decision boundary would be more flexible compared with the linear discriminant analysis.

K-nearest neighbors

K-nearest neighbors (KNN) is another supervised classification method that may be suitable for flight phase identification. The flight phase label for a given timestamp is predicted by looking at the flight phase labels from its k nearest neighbors. If most of the neighbors have the flight phase label i, then this label is assigned to the timestamp. Figure 2.5 is an example illustrated the concept of KNN in a two-dimensional space. If $k = 3$, then the red diamond point is assigned to class B, while if $k = 6$, it is assigned to class A. This method can be interpreted as using the training data as a knowledge database for predicting the newly observed data points. It is a non-parametric approach since it does not need to estimate the parameters for a model or a distribution. For smaller k values, the decision boundary of the KNN may be highly nonlinear and irregular.

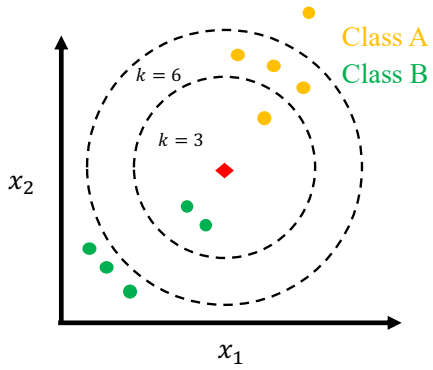


Figure 2.5: An example of KNN method in a two-dimensional space

Decision tree and random forest

The decision tree is a classifier in supervised learning, and it can be interpreted as a model with multiple layers of filters. In the method, several binary splits are applied to the feature space for creating several manifolds that correspond to data labels. An example to illustrate the concept in a two-dimensional space is shown in Figure 2.6. In this case, the feature space is separated into five distinct regions in four binary splits. The location of the split is determined by minimizing the difference between the predicted and the true labels. There

are two stopping criteria for the binary split: (1) a threshold for limiting the least amount of data points in the terminal nodes and (2) a threshold for limiting the maximum number of the terminal nodes for a tree. If a decision tree model is built to fit the training data well, this model will tend to have a large number of binary splits, and it can also have the issue of overfitting. Two fixes were proposed to tackle the overfitting problem. One is to use bagging, the acronym of "bootstrap aggregation", to reduce the variance of a fully grown decision tree. In the bagging, several candidate trees are generated in the bootstrap step while the results from these trees are combined and averaged in the aggregation step. The other approach is called the random forest, and it consists of several tree models in which binary splits are performed in a subspace of the feature space. In such a case, individual trees in the forest would look dissimilar, and thus it reduces the correlation between candidate trees. In this study, both a tree model and a random forest model are considered the candidate methods for the task of the phases of flight identification.

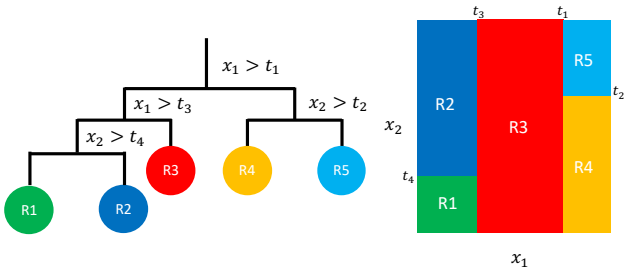


Figure 2.6: A 2D example of binary splits in a decision tree model (adapted from [75])

2.2.3 Takeoff identification

In this study, the takeoff phase is treated as a transitional phase from low-altitude to high altitude region, and it is not included in the basic phases of flight identification. However, once the basic flight phases are identified, the takeoff phase can be obtained from the information of existing flight phase labels and additional flight parameters. There are four different types of takeoffs specified in [16], namely, normal takeoff, takeoff from hover, maximum performance takeoff, and rolling takeoff. We can find several descriptions of the

takeoff phase from different sources. For example, the takeoff phase has to go through two sub-phases: the transverse flow effect, which typically happens around 10-20 knots in airspeed, and the effective translation lift (ETL), which normally occurs around 16-24 knots in airspeed. From the flight manual of the EC-135 helicopter, suggested airspeeds upon reaching certain altitudes can be found in the recommended takeoff procedure. However, it is hard to determine the start and the end of takeoff from these descriptions.

An approach attempted to tackle the takeoff identification is proposed. It is based on the following input variables: altitude, ground speed, flight path angle, weight on wheels, and flight phase labels. The first step is to estimate the liftoff point from the weight on wheels signal, which is regarded as the start of the takeoff. The end of the takeoff is determined as the timestamp where the altitude signal reaches 150 feet above ground level. If the ground speed is higher than 10 knots ahead of the liftoff point, it means that the vehicle has already begun to accelerate on the ground; thus, this segment is tagged as "rolling takeoff". If there exists a hovering phase after the liftoff point, then the segment is identified as "takeoff from hover". We use two statistical features extracted from the flight path angle to differentiate the normal and maximum performance takeoffs. A support vector machine (SVM) is trained on these features, and this classifier can be used to predict these two takeoffs. A pseudo-code detailed the procedure of this takeoff identification is documented in Appendix A.4.

2.2.4 Evaluation criteria

Since no flight phase label is attached to the flight data records used in the study, it is challenging to determine the algorithm's accuracy and effectiveness. Aside from creating a labeled dataset for addressing this difficulty, we developed some qualitative measures to evaluate the results from the algorithm. Here are some criteria that we proposed to judge the results from an algorithm for detecting flight phases:

1. Mutually exclusive: every single timestamp in the flight data records should only

belong to one flight phase.

2. Exhaustive: no unidentified timestamp should exist in the result.

3. Minimum duration: for each flight phase identified from the algorithm, the corresponding flight segment should have a minimum duration. If there exist some flight phases that only persist for less than a second, then the flight phase labels for these flight segments are required to be re-assigned.

4. Proper transition: only the practical transitions between flight phases are permissible. For instance, a climb cannot be a precursory flight phase of a takeoff, or a standing phase cannot be a successor of a cruise phase.

5. Time for computation: the algorithm is required to be efficient when dealing with flight data records in a larger sample size.

2.3 Experiments for phases of flight identification

In section 2.2, potential methods that are feasible to detect phases of flight for rotorcraft operations were introduced and presented. To find the most appropriate method or a combination of methods among the ones considered in the study, experiments that can assist in selecting the method are required. Because the flight phases are separated into two distinct regions, namely, the high and the low-altitude flight phases, the experiments will be conducted individually. For detecting the flight phases in the high-altitude region, all the methods mentioned in section 2.2.2 will be compared and tested. However, we will directly apply the filters from our baseline definitions for the flight phases in the low-altitude region due to the availability of well-defined thresholds. In the test, if unidentified data points exist, we will modify the corresponding thresholds accordingly.

2.3.1 A test on the methods for detecting the high-altitude flight phases

The experiment conducted in this subsection aims to find the best method or a combination of methods to identify phases of flight in the high-altitude region. To compare different methods quantitatively, samples of flight segments with flight phase labels attached are essential for fulfilling the task. Without the information of flight phase labels, it is not easy to make a quantitative comparison, and the performance of each method can only be judged qualitatively. Thus, to acquire the sample data needed for the experiment, we retrieved 21 flight segments in the high-altitude region from flight data records of two actual flights, and the flight phases are manually labeled using the following two strategies: (1) three dimensional flight data replay using Simulink with the integration of Flightgear and (2) animation of traces from relevant flight parameters (shown in Figure 2.7).

3D Flight Data Replay **Flight data traces animation**

Figure 2.7: Different strategies from retrieving the flight phase information

From the flight data replay, the information of various vehicle representations and the view on cockpit gauges can assist subject matter experts and pilots in determining the flight phases for the flight data records. However, as the number of flight data grows, this reviewing process would become time-consuming. Further, it is difficult to have an agreement from a group of SMEs on the exact time when a particular flight phase initiates or ends. As a result, we will first tag the flight phase labels based on both flight replay and the animation. Then, we submitted the results to a few SMEs and researchers for review

and verification.

The threshold value in the altitude for slicing the flight data records into the low and the high-altitude regions is adjustable, and it can vary in the range from 100 to 200 feet AGL. This value depends on the pattern observed in the altitude, and it is intended to minimize the total count of the low and the high-altitude samples. Given the existence of fluctuating altitude data below 200 feet, this flexible threshold can avoid consecutive flipping between flight phases in the low and the high-altitude flight phases. In our dataset, the available flight parameters include the position of the vehicle(the latitude, the longitude, and the altitude), horizontal and vertical speeds, angular position and rates. These parameters are considered as predictors when building a model in supervised learning.

In the test, the methods for capturing the high-altitude flight phases are grouped into two classes: (1) the methods can be used directly without training and (2) the methods dependent on training data. Due to no training being required for the methods in the first class, we can directly use all the samples for testing. If hyperparameters need to be tuned for specific methods, the values are selected by minimizing the prediction error. For the methods in supervised learning, the labeled dataset have to be split into two sets: one for training and the other for testing. To train the models, we implemented 10-fold cross-validation for choosing the optimal model parameters. Some details of the settings for each of the methods considered are described as follows:

1. *Filtering approach*: The thresholds on flight parameters for detecting the high-altitude flight phases are summarized in Table 2.4. If the climb rate is higher than 90 feet per minute, the corresponding timestamps are classified as a climb. If the descent rate is higher than 90 feet/min, the flight segment is identified in a descent phase. For the remaining data points, a cruise or a high-altitude hover could be assigned depending on whether the ground speed is higher or lower than 0.6 knots.

2. *Piecewise linear regression*: The goodness-of-fit parameter is set to 0.99, and the slope threshold to differentiate the climb, the cruise, and the descent is 90 feet per

minute, which is the same as the filtering approach. If the sequence smoother is used in conjunction with the PLR, the smallest window size is selected to be 10 seconds.

3. *Sliding window regression classification*: Two tuning parameters are associated with the SWRC: the size of the sliding window and the slope threshold. An optimization is performed for these tuning parameters, and the ranges of window size and slope are selected as 10-200 (equivalent to 2.5-50 sec) and 1-5 (equivalent to 60-300 feet/min), respectively. The optimal values for the window size and slope threshold are 140 and 2.5, separately.

4. *Logistic regression*: The idea of using different combinations of flight parameters as the predictors for the model was tested. As a result, the altitude, horizontal and vertical speeds were chosen as the predictors.

5. *Naive Bayes classifier*: The flight parameters selected to be the predictors for the Naive Bayes classifier are the same as those used for the logistic regression. The conditional distribution of a flight parameter given being in a particular flight phase is assumed to follow a normal distribution.

6. *Linear and quadratic discriminant analyses*: The flight parameters selected to be the predictors for the discriminant analyses are the same as those used for the logistic regression.

7. *K-nearest neighbors*: The features decided on the KNN model are identical to the methods mentioned above. To find the optimal value of k, we examined the profiles of training and testing errors. The best k value is determined by balancing these two types of errors, and the value is set as 15 in the test.

8. *Tree model and random forest*: The number of splits controls the model complexity of a tree model. Based on the cross-validation result, the optimal value is set as 20 in

the test. For the hyperparameters involved in the random forest model, the following values are chosen:

- In-bag fraction of samples = 0.9
- Number of predictors = 7
- Number of bagged trees = 10
- Maximum number of splits = 10

The classification results for the methods no need for training are displayed in Figure 2.8. On the whole, the piecewise linear regression with sequence smoother and the sliding window regression classification have lower prediction errors than the other two in the class. If we only consider the PLR with sequence smoother and the SWRC, the PLR with sequence smoother did outperform the SWRC in certain cases. Nonetheless, the SWRC has the lowest classification error on average in this experiment.

To dive further into the detailed difference between individual methods, we selected the results from a sample flight segment, and they are presented in Figure 2.9 and 2.10. The subplots on the top row show the actual flight phase labels in each figure, while the subplots on the bottom display the predicted flight phase labels from the methods. In Figure 2.9, both the PLR and the approach based on filters are suffered from having flight phases in short duration and consecutive switches between flight phases. Another issue for the filtering method is that the cruise phase would exist as an intermediate phase when going from a climb to descent or in reverse order because vertical speed is a continuous signal. In the actual scenario, a helicopter flight can go directly from the climb to the descent without passing through a cruise segment. The piecewise linear regression alone can capture the flight phases when the altitude data exhibit a smooth pattern. In cases like having fluctuation in a cruise phase, the PLR would label the bumps and dips in altitude as individual climbs and descents rather than a whole piece of the cruise. The sequence smoother is a tool designed to lower the impact when dealing with altitude signals with

local fluctuations, and it can assist in eliminating the flight phases with short duration for the PLR method. In Figure 2.10, the results from PLR with sequence smoother and the SWRC are presented. Although having a minute difference in specific local regions, most of the flight phases detected using these two methods agree with the true flight phase labels.

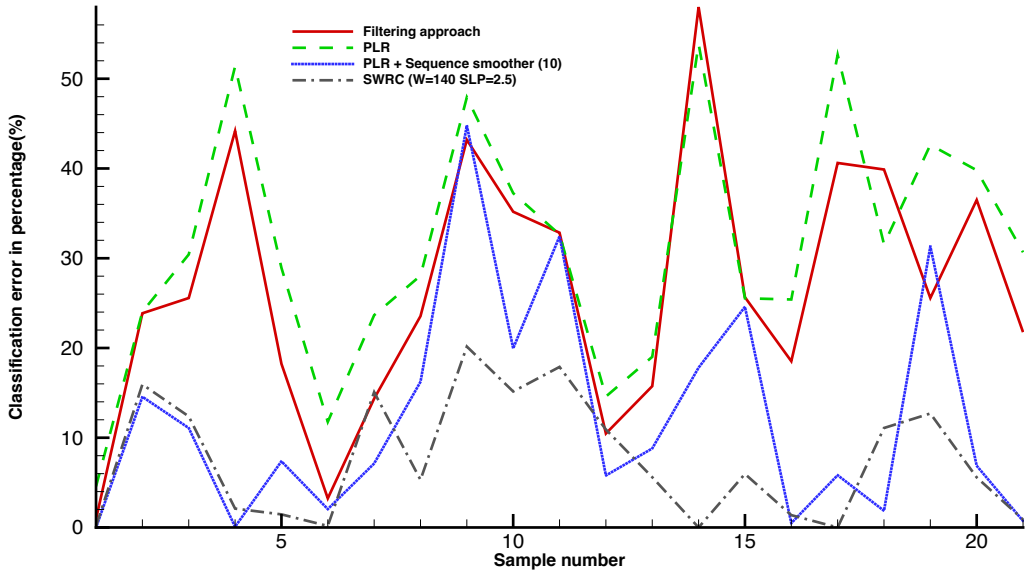


Figure 2.8: The comparison of the methods no need for training

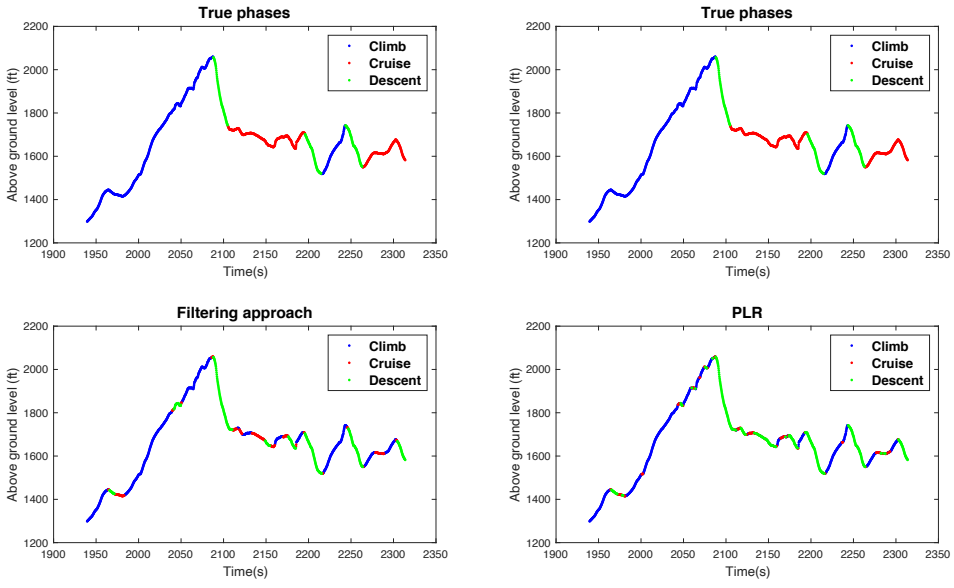


Figure 2.9: Sample results from the filtering approach and the PLR

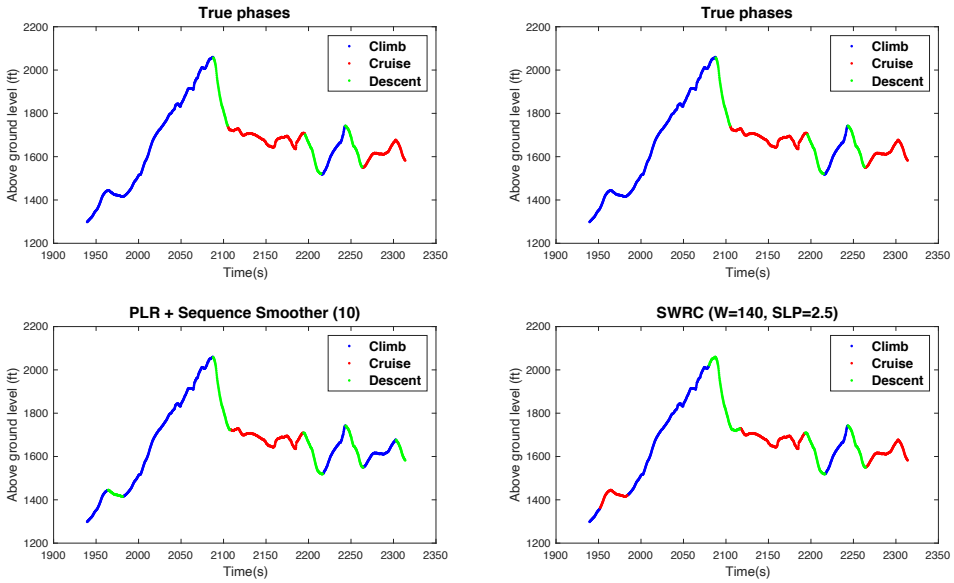


Figure 2.10: Sample results from the PLR with sequence smoother and the SWRC

In Figure 2.11, the identification results from the methods in the second class, which correspond to the ones in supervised learning, are presented. It is observed that these methods fulfill the task on samples such as #1, #5, #6, and #10, but they have low accuracy in prediction for samples #2 and #7. Overall, the logistic regression with three predictors has the highest accuracy in prediction among the methods in this class. Two potential root causes can contribute to the inconsistency of prediction accuracy across different samples. First, the data points being fed into the models in supervised learning are typically assumed to be independent. However, the flight parameters are in the form of time series, and the neighboring data points may not be independent. Second, these models were trained under a relatively small dataset and thus may not be sufficient to work in various scenarios. With more labeled data acquired in the future, the performance of these methods is expected to improve. To sum up, an overall comparison among all methods considered is shown in Figure 2.12. The PLR with smoother sequence and sliding window regression classification achieve higher prediction accuracy than others. Therefore, they are selected to identify the high-altitude flight phases.

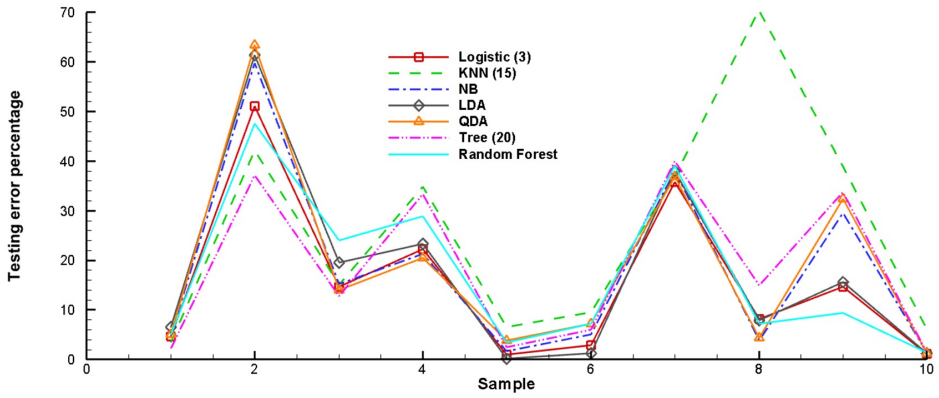


Figure 2.11: The comparison of the methods dependent on training data

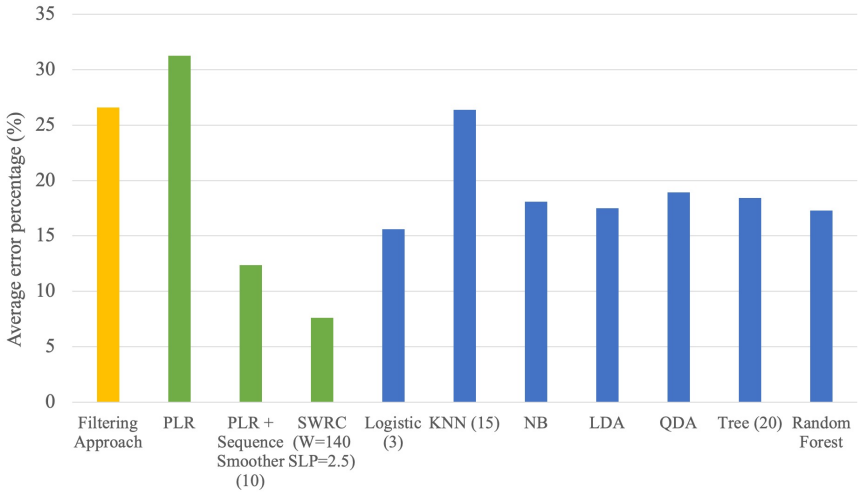


Figure 2.12: Overall comparison of all methods considered for detecting flight phases

2.3.2 A test on the filters for detecting the low-altitude flight phases

For the low-altitude flight phases, quantitative thresholds on flight parameters can be found in our baseline definitions in Table 2.4. Without the actual flight phase labels attached to the data, the results from the filtering approach will be qualitatively evaluated using the criteria proposed in section 2.2.4. If the results cannot meet the requirements, we will modify the previously selected threshold values. In the test, we found unidentified data points for the flight data records examined, which violates the criterion of exhaustivity. Thus, the following changes are made to the thresholds for detecting the low-altitude flight phases:

- *Increase the upper bound of the ground speed threshold for the air taxi*: It was observed that some flight segments belonging to the air taxi phase would have a higher ground speed than the upper bound specified in our baseline definition.

- *Increase the ground speed threshold for distinguishing the standing and the hover from taxi phases*: A higher ground speed threshold can alleviate the frequent flips between the flight phases with and without motion to account for noisy ground speed signals.

In Figure 2.13, threshold values on flight parameters for detecting the low-altitude flight phases are displayed as colored blocks in three-dimensional feature space. The left subplot in the figure is the constraints specified in the baseline definitions, while the right subplot shows the modified constraints for addressing the issue of unidentified data points. In the visualization, the range of the vertical speed is from -200 to 200 feet per minute, and if the blocks are in contact with these boundaries, it means that no upper or lower bound is placed on the vertical speed for these flight phases. In the test of using modified constraints, all the results are satisfied with the proposed evaluation criteria. A handful of sample results of using the filtering approach to detect the low-altitude flight phases are demonstrated in Figure 2.14. From the first sample shown on the top subplot, the aircraft is prepared to take off, and the flight phases for this flight segment are switching between the hover and the air taxi. For the second sample displayed on the bottom subplot, the aircraft is coming from an approach. After lingering at the low-altitude region for a moment, it is ready to take off again. We can see more low-altitude flight phases are involved in this particular flight segment. In the future, it is envisioned that these constraints might need to be adjusted based on the operations, and more flight parameters may be added to the feature space for better capturing the characteristics of flight phases.

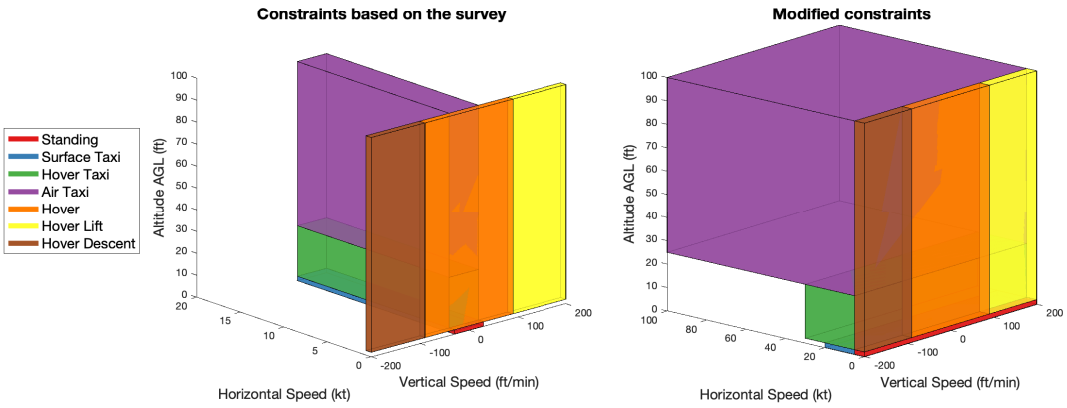


Figure 2.13: Visualization of thresholds on flight parameters for detecting flight phases

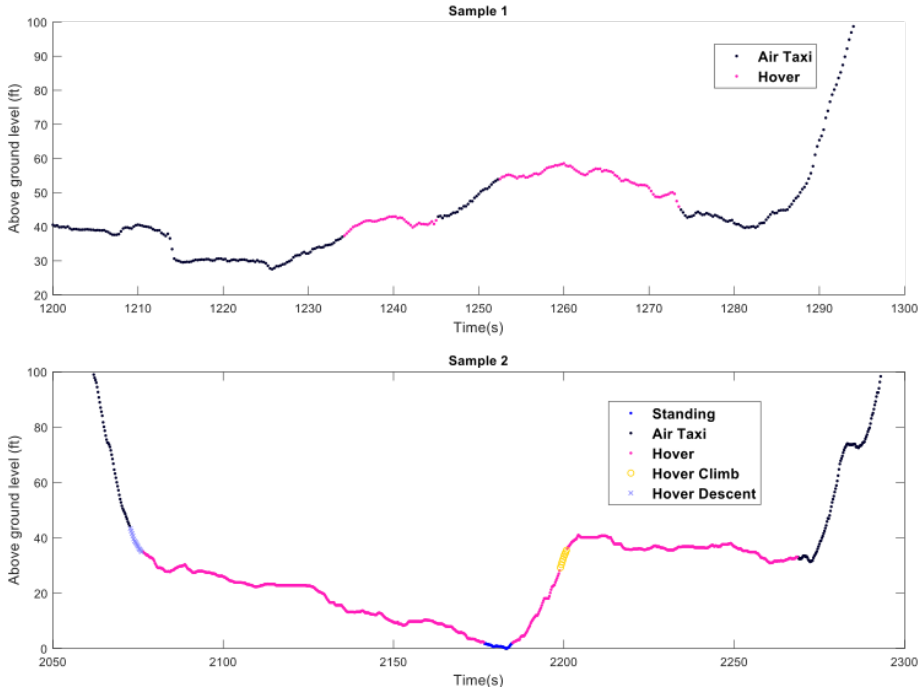


Figure 2.14: Sample results of using the filtering approach for detecting the low-altitude flight phases

2.3.3 Add-ons to the basic phases of flight identification algorithm

We can add several additional flight phases to the basic phases of the flight identification algorithm. For example, the turn maneuver can be detected using the same method for differentiating the climb, the cruise, and the descent phases. By switching the flight parameter from the altitude to the heading, the piecewise linear regression (PLR) with sequence

smoother can detect the phases of the right turn, left turn, and stay straight. A sample result is shown in Figure 2.15, and it is obvious to see that multiple turn events are accurately captured.

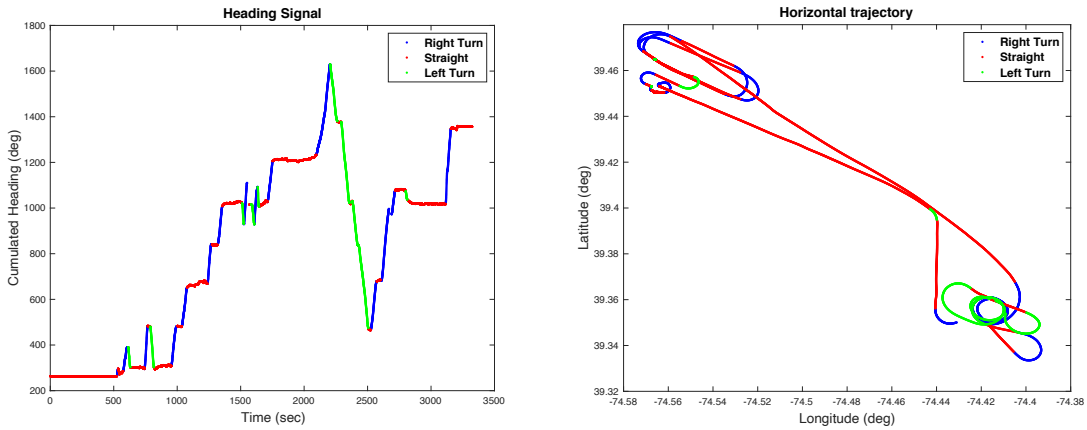


Figure 2.15: Results of turn detection

Another flight phase that can be incorporated into the basic phases of flight identification algorithm is takeoff. As mentioned in section 2.2.3, the takeoff is treated as a transition phase from the low to the high-altitude region. Once the timestamps corresponding to this transition are identified, we can assign different takeoff labels based on relevant flight parameters and flight phase labels in the duration. We have prototyped an algorithm for detecting the takeoff, and it is intended to use simulated runs for algorithm verification. These simulated runs were conducted using an X-Plane simulation environment, and an observer in the cockpit would document all the takeoffs. A snapshot of camera views from the cockpit for different takeoff maneuvers is shown in Figure 2.16. In the test, we used the first few batches of simulated runs to tune the parameters in the algorithm, and the remaining simulated data were managed to evaluate the algorithm's performance. In Table 2.5, the number of simulated takeoffs in each category available for verification is provided. From the results of this test, the algorithm performs well in detecting various takeoff types, especially for the takeoff from hover and the rolling takeoff. For improving the accuracy of differentiating the normal and the maximum performance takeoffs, we can add more new

labeled data to the training process of the SVM classifier, and the resulting new decision boundary can be used to distinguish these two takeoffs more precisely.

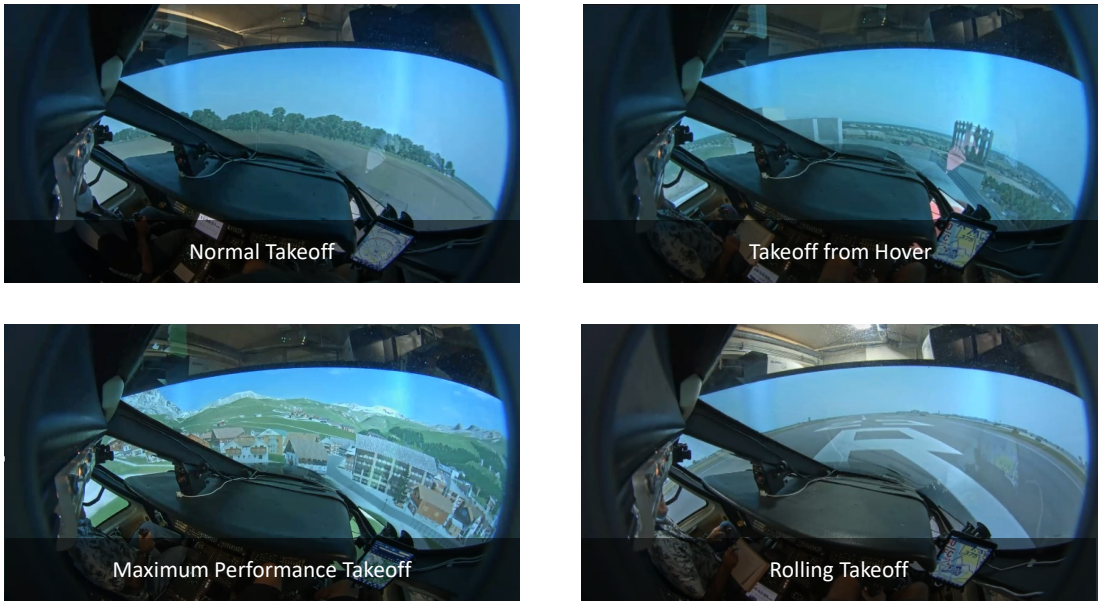


Figure 2.16: Different takeoff maneuvers conducted in a X-Plane simulation environment

Table 2.5: Results of takeoff identification using simulator

Summary of Takeoff Simulation				
	Normal Takeoff	**Takeoff from Hover**	**Maximum Performance Takeoff**	**Rolling Takeoff**
Number of simulated segments	21	15	10	18
Number of correctly identified	19	15	8	18

2.4 Summary for the second research question

To move forward in our first research objective, we first addressed the need to define the phases of flight for rotorcraft operations. Since flight phase identification is a prerequisite and a pathway that leads to anomaly detection, a methodology that can assist in detecting flight phases was proposed. Several methods are considered for the task, including a filtering approach, regression-based classifiers, and supervised learning methods. In the high-altitude flight phases test, the piecewise linear regression (PLR) with sequence smoother

and the sliding window regression classification (SWRC) are the best among the methods considered in terms of average accuracy across all labeled samples. We selected a filtering approach with modified constraints in the low-altitude flight phases test due to the well-defined thresholds and its simplicity. The combined identification results from these two regions meet the evaluation criteria that we proposed for judging the algorithm's performance. At last, an implementation to detect the takeoff, which is a flight phase initially treated as a transitional flight phase, was presented. This add-on logic was verified using simulated trials, and it is capable of accurately predicting different types of takeoff.

CHAPTER 3
ANOMALY DETECTION ON FLIGHT DATA RECORDS

In chapter 2, a baseline algorithm is established for detecting the phases of flight for rotorcraft operations. Before proceeding to the task of anomaly detection, it is essential to have a definition of the anomaly we aim to catch. The overarching objective here is to leverage data mining techniques and use patterns discovered for identifying the norm and the outliers in a specific group of flight segments. Several methods that can assist the task will be covered in this chapter, and we also set up a few experiments to test their performance. In the end, we will apply the best methods among the ones considered to actual flight segments for validating the selection.

3.1 Problem formulation

Once we acquired the phases of flight from flight data records, a statistical analysis on this information can be used to characterize a fleet-level of flights. Moreover, the task of anomaly detection can be performed on flight segments within the same flight phase. The exceedance analysis is the traditional approach for finding anomalies in flight data records. However, the threshold values for exceedances need to be determined either by the subject matter experts (SME) or by the manufacturers before the analysis, and they can vary with mission profiles and the types of helicopters. To avoid the step of threshold determination, we adopted a data-driven approach in this study to detect anomalies by examining a sufficient amount of flight data records. The third research question relevant to the task of anomaly detection is stated as follows:

RQ3: Given that the flight segments belong to the same flight phase and know the type of anomaly to detect, what techniques can be applied to discover patterns such that anomaly detection can be performed without the availability of predefined thresholds?

The hypothesis associated with the third research question is

H3: Due to the unavailability of a labeled dataset, the techniques belonging to unsupervised learning are chosen. A sequential approach that contains feature extraction and clustering analysis is proposed to tackle both (1) time series data and (2) trajectory data for assisting the task of anomaly detection.

To test the third hypothesis, we have designed some relevant experiments, and they are described as follows:

E3: Different combinations of feature extraction and clustering techniques will be first tested using synthetic and simulated data. The ones capable of capturing the patterns of both types of data will then be applied to the real dataset.

In the following approach section, the type of anomalies to detect is first defined, and then an overview of the process is presented. The methods used in each step of the process will be explained as well.

3.2 Approach for anomaly detection on flight data records

Before diving into various methods that may be useful for anomaly detection in flight data records, it is essential to have a proper definition of an "anomaly". An anomaly, sometimes referred to as an outlier or a novelty, is an entity that is rarely observed and has a different behavior or pattern compared to the majority or the normality within a group.

Based on the work by Chandola et al. [76], three crucial kinds of anomalies, namely point anomalies, contextual anomalies, and collective anomalies, can be potentially found with regard to time series. In this study, our focus will be placed on developing a framework for detecting collective anomalies, in which time series are treated as a whole rather than independent sequential observations. It is important to note that anomalies detected using this framework may not directly correspond to hazardous events. Instead, the segments being tagged as "anomalous" are better interpreted as being "rare" and "dissimilar" from the rest of the group. Further, flight segments being investigated in the anomaly detection should be within the same phase of flight. The extraction of homogenous flight segments can be performed using the method mentioned in section 2.2 or other algorithms when applicable.

There are some challenges associated with detecting anomalous flight segments in flight data records we currently have. First, subject matter experts have not investigated the flights present in the database. No anomalous tag has been placed on an entire flight or specific timestamps of particular flights. Therefore, supervised learning methods, often used for classification and anomaly detection, cannot be applied. Instead, unsupervised learning methods may be helpful in this situation. It is generally harder to handle the unsupervised learning problem due to the difficulty of evaluating the model performance without labels. Second, the standard length units for flight parameters are time and nautical mile, and the flight segments are often in unequal lengths. To reasonably compare homogenous flight segments, a procedure needs to be developed. The methods that we used to address these challenges are explained in the following subsections.

To detect anomalies in routine flight data records for helicopter operations, a sequential approach, which consists of multiple steps, is proposed and is depicted in Figure 3.1. Prior to the final outlier detection step, several filtering steps are implemented to eliminate inconsistent segments and make a fair comparison. The approach starts by grouping heterogeneous flight data records based on contextual notions, such as the type of operations and the make and model of rotorcraft. This step ensures that the flights within a given group are

similar and removes variations between flights due to such categorical factors. Then, the flights are segmented into comparable pieces, i.e., phases of flight, so that the subsequent analyses can be conducted on the same fundamental level. Past research on anomaly detection in aviation data showed that the proposed analyses are typically performed on flight segments within the same flight phase rather than the entire flight data records. Therefore, a flight phase identification algorithm for rotorcraft is a prerequisite for the task of anomaly detection.

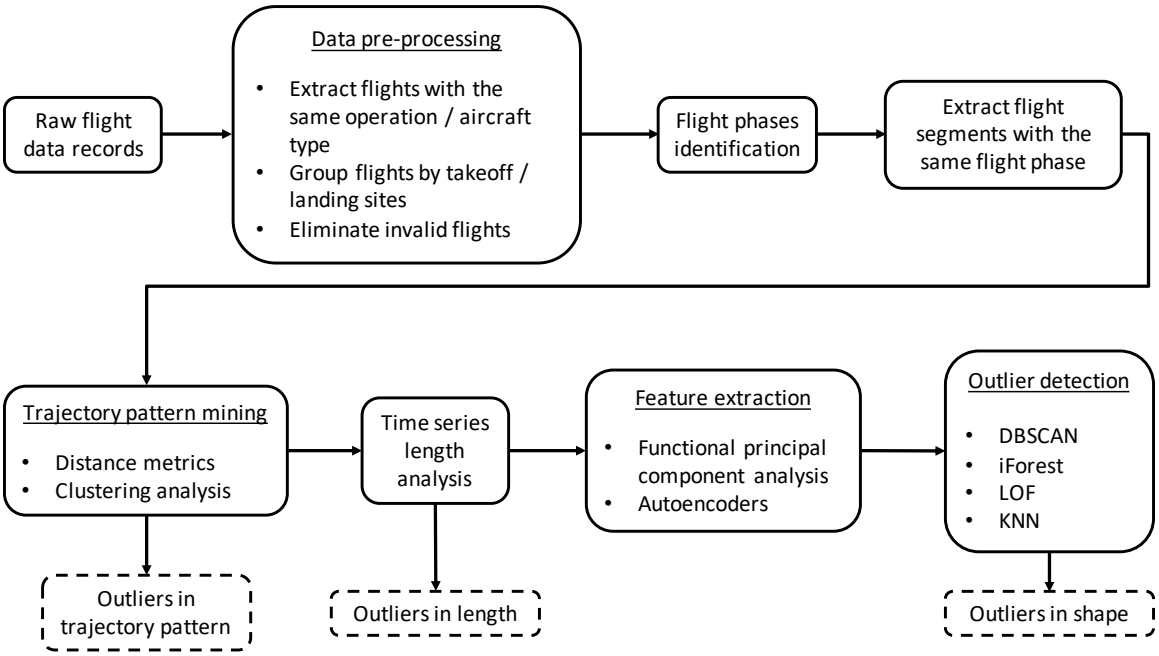


Figure 3.1: General framework for anomaly detection of helicopter flight data records

Trajectory pattern mining is used to associate flight segments with similar spatial paths after the flight phase of interest is identified for each flight data record considered. In this step, we adopted some distance measures used in the literature to calculate the similarity between trajectories. The results are then served as the inputs to the clustering analysis for pattern recognition. Segments that do not follow the primary spatial pattern are marked as outliers, and the remaining segments are further examined in terms of the flight parameters involved. These parameters may include native variables like horizontal and vertical speeds and derived variables such as specific energy and flight path angle. To address the challenge

of segments in unequal lengths, we characterized these segments by the corresponding length and shape features. With the availability of these two features, the profile of the signal in its original domain can be recovered. Some efforts have been put into combining the length and shape features as one set of features by concatenation or by using feature scaling/shifting. However, we found two drawbacks to blending those features. The first one is the difficulty of recognizing why a data point is detected as an outlier. The second one is that, in some extreme cases, even if two different time series look drastically dissimilar in shape, they can still be close to one another in the feature space due to the inappropriate scaling/shifting of the shape features. For example, in the left subplot of Figure 3.2, the shape information of two sets of curves (represented by red and blue circles) is captured with features in a two-dimensional space. The curves represented by red circles have a longer duration than the curves represented by blue circles, and the histogram of segment duration is shown in the middle subplot. When trying to blend the length information with the shape features using shifting (shown in the left subplot), these two sets of curves cannot be differentiated in the blended feature space.

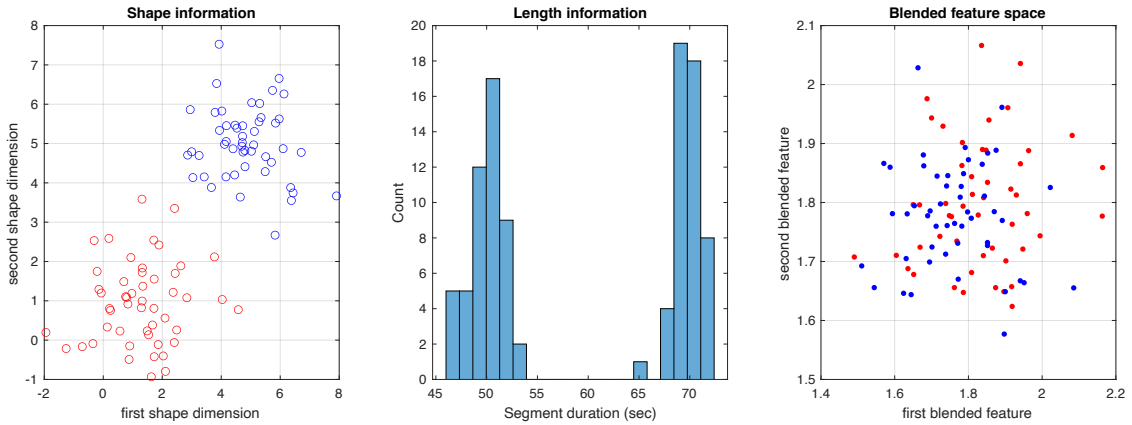


Figure 3.2: An example of blending shape and length features

Therefore, a two-step approach is taken to tackle segments in unequal lengths. First, outliers in length are detected if the segment length falls outside of the designed range $[l_{\text{nominal}} - 30 \text{ sec}, l_{\text{nominal}} + 30 \text{ sec}]$, where l_{nominal} is the nominal length, selected as the most

observed length or the median length over all the segments considered. Second, the shape information is extracted for all the segments of comparable lengths, and outlier detection is performed on the shape features. Ultimately, three distinct types of outliers may be detected in the process. To obtain the shape information out of time series, we considered several feature extraction methods, including functional principal component analysis (FPCA) and various autoencoders (AE), and they are examined and tested. Combinations of these feature extraction methods and several outlier detection methods form a portfolio of methods that may be helpful to identify shape anomalies. They are shown in Figure 3.3 and a brief introduction of each method is provided in the next subsections.

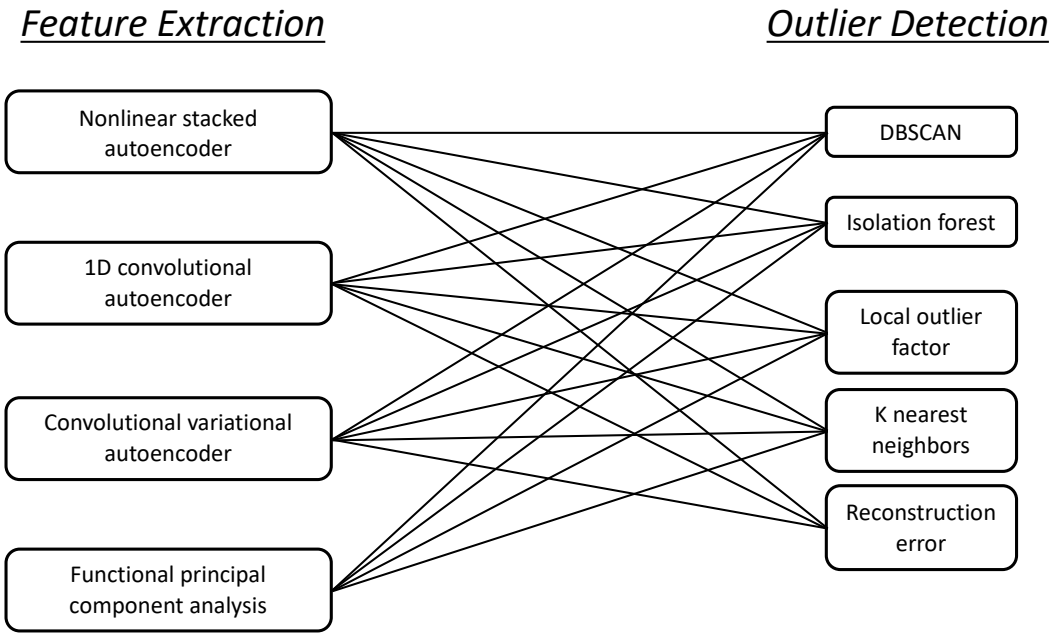


Figure 3.3: The methods considered for detecting shape anomalies in time series

Finally, due to the lack of actual labels in the real dataset, it is hard to evaluate the performance of the methods mentioned above in the portfolio and pick the most suitable combination. To address this issue, synthetic and simulated data are used to test the effectiveness of the methods in the portfolio. It is hypothesized that if there exists a similarity between synthetic/simulated data and real data, then the methods appropriate for the synthetic/simulated data should also be applicable to the real data. The details of how to create

synthetic/simulated data and the way they are used for selecting candidate methods are described in the experiments section.

3.2.1 Methods for trajectory pattern mining

To detect patterns in flight trajectories, distance metrics are applied to trajectories for comparing the similarity between different geometric shapes. Then clustering analysis is performed for placing trajectories into relevant groups. There are several distance metrics applicable to trajectory comparison, and they will be described later. It should be noted that trajectories within the same flight phase are typically not the same length. If the selected distance metrics cannot handle segments in unequal lengths, the trajectories considered should be transformed into consistent lengths using curve interpolation. After calculating distances between distinct trajectories, a clustering algorithm that can take the pairwise distance matrix as its input argument is applied to discern patterns in trajectories. Here we choose hierarchical clustering given its simple implementation and interpretation. The dendrogram, which is a typical result of hierarchical clustering, shows how clusters are formed and combined based on the selected linkage. The number of clusters is determined by selecting a cutoff value for the height of the dendrogram. In general, larger cutoff values would produce fewer clusters. In this study, the cutoff value was set as the upper extreme of the Box-Whisker plot for all pairwise distances of trajectories. Through testing a few sample sets with known labels, this cutoff value allows us to find the valid number of clusters.

Euclidean distance

Suppose there exist two trajectories $\mathbf{x}$ and $\mathbf{y}$ of the same length n. The Euclidean distance between these two trajectories is defined as

$$d_{\text{ED}}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3.1}$$

where $\mathbf{x} = (x_1, \cdots, x_i, \cdots, x_n)$ and $\mathbf{y} = (y_1, \cdots, y_i, \cdots, y_n)$. Trajectories of unequal lengths may be tailored to the same length using curve interpolation under uniformly spaced points over a chosen interval. To preserve the shape characteristics of the trajectories, the number of points used for the interpolation should be large enough in order to capture minute or local variations in the trajectories.

Hausdorff distance

The Hausdorff distance is a measure used to find the dissimilarity between two sets of points. Assume the first trajectory $X = \{x_1, \cdots, x_m\}$ consists of m waypoints, and the second trajectory $Y = \{y_1, \cdots, y_n\}$ has n waypoints, where m is not necessarily equal to n. The directed or one-sided Hausdorff distance is defined as

$$\tilde{d}_{\text{HD}}(X, Y) = \max_{x \in X} \min_{y \in Y} \|x - y\| \tag{3.2}$$

$\tilde{d}_{\text{HD}}$ can be interpreted as first finding the distance of the nearest point in the set Y for every point in the set X, and then, taking the maximum as the representative distance between the two sets. It is not a symmetric metric and a more general Hausdorff distance is expressed as follows:

$$d_{\text{HD}}(X, Y) = \max(\tilde{d}_{\text{HD}}(X, Y), \tilde{d}_{\text{HD}}(Y, X)) \tag{3.3}$$

It can be observed that the Hausdorff distance treats waypoints as independent elements in a set and does not take the order of the waypoints into account. For some edge cases, the Hausdorff distance can be small while the shapes of the trajectories look drastically different.

Dynamic time warping

Dynamic time warping (DTW) is a distance measure designed to deal with when two trajectories have similar shapes but are not in sync in the time axis. If the Euclidean distance

is applied to this scenario, the dissimilarity between these two trajectories would be higher compared to the case using dynamic time warping. Due to its "time-warping" characteristics, a waypoint in one trajectory is not restricted to be aligned with its counterpart from another trajectory in the same timestamp. It can be aligned with other waypoints in the adjacent timestamps of another trajectory, and thus this measure is suitable for comparing asynchronous trajectories. To perform dynamic time warping, a grid map that contains the pairwise distances of waypoints in two trajectories is first constructed. The alignment of the trajectories is obtained by finding the shortest path from the lower-left corner of the map to the upper-right corner, and this problem can be solved using a dynamic programming algorithm. Different constraints are often added to the optimization formulation to reduce the number of viable paths in the search space. A cartoon that explains the concept of dynamic time warping is shown in Figure 3.4.

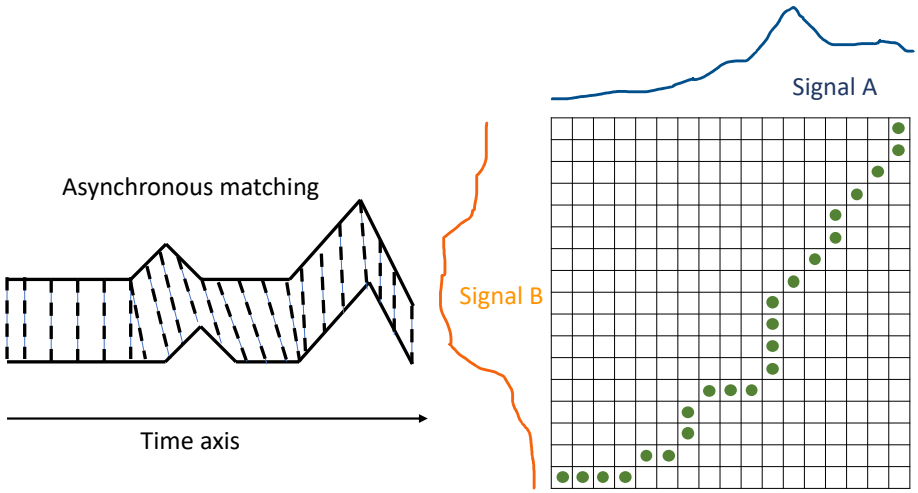


Figure 3.4: An illustration of dynamic time warping algorithm

3.2.2 Methods for extracting features from time series

To detect unusual or rare shapes in time series, the representative features are required to be extracted before performing the outlier detection. These features should capture the essence of the original time series in a parsimonious fashion. Two valuable methods, functional

principal component analysis, and autoencoders are introduced as follows.

Functional principal component analysis

Assume the observed time series can be decomposed into the mean and the residual shown in Equation (3.4)

$$S_i(t) = \mu(t) + \epsilon_i(t) \tag{3.4}$$

where $S_i(t)$ is the i-th observed signal, $\mu(t)$ is the mean signal, and $\epsilon_i(t)$ is the residual of the i-th observed signal. Through Karhunen-Loève theorem, the residuals can be represented as an infinite sum of the products between functional principal component (FPC) scores and their corresponding eigenfunctions, which is expressed in Equation (3.5)

$$\epsilon_i(t) = \sum_{k=1}^{\infty} \xi_{ik}\phi_k(t) \tag{3.5}$$

where ξ_{ik} is the k-th FPC score for i-th signal and $\phi_k(t)$ is the k-th eigenfunctions. In functional principal component analysis (FPCA), a continuous curve or profile, which is essentially an infinite-dimensional entity, can be approximately represented by several FPC scores through the eigen-decomposition of the covariance function. The eigen-decomposition is shown in Equation (3.6)

$$\int_0^M \hat{C}(t,t')\hat{\phi}_k(t)dt = \hat{\lambda}_k\hat{\phi}_k(t') \tag{3.6}$$

where $\hat{C}(t,t')$ is the estimated covariance function, $\hat{\lambda}_k$ is the k-th eigenvalue, and $\hat{\phi}_k(t)$ is the eigenfunction corresponding to the k-th eigenvalue. With the estimated eigenfunction $\hat{\phi}_k(t)$, the k-th FPC score for any observed curves can be calculated using Equation (3.7)

$$\xi_{ik} = \int_0^M \epsilon_i(t)\hat{\phi}_k(t)dt \tag{3.7}$$

The original signal $S_i(t)$ can be approximately recovered using K's FPC scores where K is

the number of FPC scores for explaining 95% of the variance observed. The approximation of $S_i(t)$ is shown in Equation (3.8)

$$S_i(t) \approx \mu(t) + \sum_{k=1}^{K} \xi_{ik}\hat{\phi}_k(t) \tag{3.8}$$

The ξ_{ik}'s are used as the low-dimensional representation of the time series data. It is observed that under a fixed percentage variance explained, higher the variation of the signal would lead to higher the dimension for the FPC scores.

Autoencoders

An autoencoder is a particular type of neural network which compresses input signals to a lower-dimensional representation and outputs the decompressed signals resembling the inputs. Given that the aim is to match the inputs with the outputs rather than to compare the predicted labels with the true labels, it is an unsupervised learning method, and no label is required to train the model's coefficients. It has two major components: (1) an encoder which projects a high-dimensional input to a representation in low-dimensional embedding or latent space and (2) a decoder for recovering the original signal. Suppose that $\mathbf{x}$ is the input of an autoencoder, $\mathbf{z}$ is the feature extracted from the signal, $f(\cdot)$ represents the encoder, and $g(\cdot)$ stands for the decoder. The output of the autoencoder can be written as

$$\hat{\mathbf{x}} = g(\mathbf{z}) = g(f(\mathbf{x})),\ \mathbf{x} \in \mathbb{R}^l,\ \mathbf{z} \in \mathbb{R}^m, \text{where } l \geq m \tag{3.9}$$

The weights of an autoencoder are learned through a backward propagation process in which a designed loss function is minimized. In general, the loss function would take the following form:

$$L(\Theta) = \sum_{i=1}^{n} \sqrt{(\mathbf{x}_i - \hat{\mathbf{x}}_i)^2} + \text{regularizer} \tag{3.10}$$

where Θ are the coefficients of the autoencoder and n is the number of input signals or the number of batch signals if choosing stochastic gradient descent as the optimizer. The first term is the reconstruction error in the form of the mean square error. The second term is a regularization that can serve different purposes, such as encouraging sparsity on the dimension of the encoding layer or placing constraints on weights. Autoencoders have a variety of applications ranging from dimensional reduction to image denoising and generation. In this study, we will use it for the purpose of anomaly detection. There are two potential routes that link anomaly detection with an autoencoder. One is to perform outlier detection on the low-dimensional features acquired from the encoding process. The other approach would rely on the reconstruction error to measure the tendency toward outliers. The corresponding flight segment would have a higher anomalous score with a higher reconstruction error. It is noted from previous research that in order to have autoencoders perform well for anomaly detection, the model should only be trained on normal data. For the data without labels attached, outliers are likely to be included in the training process, and the performance might deteriorate, especially for the cases with a high percentage of outliers. There are many kinds of autoencoders being studied in the literature. Here, only three types of autoencoders will be considered: a nonlinear stacked autoencoder, a one-dimensional convolutional autoencoder (1DCAE), and a convolutional variational autoencoder (CVAE).

A nonlinear stacked autoencoder is an extension of the traditional principal component analysis (PCA), which is to some extent equivalent to a single-layer, linear activated autoencoder. By introducing nonlinear activation functions such as a rectifier linear unit (ReLU) and stacking multiple layers, the extracted features in the latent space should better capture the nonlinear information of the input signals. The second autoencoder considered is a one-dimensional convolutional autoencoder (1DCAE). The architecture of 1DCAE can be interpreted as a deck of multiple dense layers wrapped by some custom layers on both sides of the encoder and the decoder. The custom layer on the encoder side typically consists of a convolutional layer plus a pooling layer. On the decoder side, the custom layer includes

a de-convolutional layer with an upsampling layer. These custom layers work in exactly the opposite manner to serve compression and decompression functionality separately. The convolutional layer in 1DCAE provides a benefit over the nonlinear stacked autoencoder by using filters to locally extract temporal information from input signals rather than dealing with points in the signal as independent observations. The last autoencoder examined is a convolutional variational autoencoder (CVAE). It is similar to 1DCAE for the convolutional part, but it has a probabilistic twist for generating features in the latent space. Instead of directly shrinking the dimension of the latent space using multiple dense layers, CVAE assumes that the features follow a specific distribution and are constructed through a sampling process from two sub-layers; one represents the mean, and the other corresponds to the variance. The loss function of CVAE contains not only the reconstruction error but also a regularization term in the form of Kullback-Leibler (KL) divergence. This KL divergence is a measure for calculating the distance between two distributions, and the KL loss encourages the distribution of the features to be closer to the assumed distribution. One key difference between CVAE and other autoencoders is that it is a generative model; thus, it can produce new sample data either by following the same pattern as the training data or other patterns not observed in the training set. For more details of the CVAE, readers may refer to [77]. For all the autoencoders mentioned above, we assume the dimension of latent space is restricted to two-dimensional space for better visualization and understanding of the structure. In this study, we will use the tool TensorFlow [78] for the implementation, and the layout of each autoencoder is documented in Appendix B.

3.2.3 Methods for outlier detection

DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a clustering algorithm that supports outlier detection. The abnormality of a data point is determined by the number of points within its neighborhood. Compared with other clustering algorithms

like k-means clustering, DBSCAN does not require the number of clusters as its input argument. Instead, two parameters are needed for DBSCAN: the minimum number of points (*minPts*) for defining a cluster, and the radius ϵ for specifying the neighborhood. A data point is classified as a core point if the number of surrounding points exceeds *minPts* within its ϵ neighborhood. If a point has fewer than *minPts* existed in its neighborhood, but it is still reachable from a core point, it is categorized as a border point. A point neither a core point nor a border point is an outlier. A graphical representation of the above description is shown in Figure 3.5. To set the value for ϵ under a selected *minPts*, k-distance graph is used and the distance where the curve has a sharp change is the optimal value for radius ϵ.

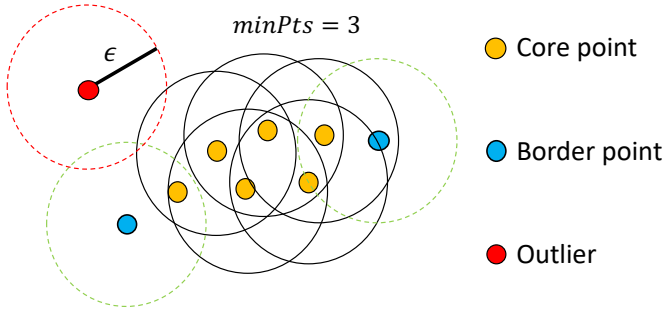


Figure 3.5: An illustration of DBSCAN clustering

Isolation forest

Isolation forest is an ensemble method developed by Liu et al. [79]. This method leveraged the idea of a binary search tree (BST). If a data point can be isolated in fewer binary splits or, in other words, easier to be isolated, then this point is more likely to be an outlier. The construction of an isolation forest starts from building an isolation tree (iTree). To form an iTree, a dimension in the feature space and a threshold value for the binary split are randomly chosen. The binary split on the feature space is repeated until every data point has been placed in a leaf node. Once the iTree is formed, every data point is associated with a path length, i.e., the distance from the root node to the corresponding leaf node. The

anomaly score for the isolation forest is then defined as follows:

$$S(\mathbf{x}, n) = 2^{-\frac{\mathbb{E}(h(\mathbf{x}))}{c(n)}} \tag{3.11}$$

where $\mathbf{x}$ is an individual data point, n is the sample size, $h(\mathbf{x})$ is the path length for data $\mathbf{x}$, and $c(n)$ is the average path length. It should be noted that the expectation that existed in the numerator of the power term is taken over all the iTree considered in the isolation forest. If the anomaly score is closer to one, it means that the path length for $\mathbf{x}$ is much shorter compared with the average path length, so $\mathbf{x}$ has a higher chance to be an outlier. If the anomaly score is approaching zero, $\mathbf{x}$ is more likely to be a normal point.

Local outlier factor

The local outlier factor (LOF) is a density-based outlier detection method. If a point is situated in a densely populated region, it is more likely to be treated as a normal point in contrast to a point located in a sparsely populated region. Assume the distance to its k-th nearest neighbor for a point $\mathbf{x}$ is denoted as $d_k(\mathbf{x})$ and the reachability distance to a point $\mathbf{y}$ can be expressed as

$$d_{\text{reach}}(\mathbf{x}, \mathbf{y}) = \max\{d_{\text{ED}}(\mathbf{x}, \mathbf{y}), d_k(\mathbf{y})\} \tag{3.12}$$

If the k closest points to the point $\mathbf{x}$ are represented as $N_k(\mathbf{x})$, the average reachability distance is

$$\bar{d}_{\text{reach}}(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \in N_k(\mathbf{x})} d_{\text{reach}}(\mathbf{x}, \mathbf{y}) \tag{3.13}$$

Given the average reachability distance, the LOF is defined as

$$LOF(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \in N_k(\mathbf{x})} \frac{1}{\bar{d}_{\text{reach}}(\mathbf{y})} \bar{d}_{\text{reach}}(\mathbf{x}) \tag{3.14}$$

The LOF can be interpreted as the normalized version of the average reachability distance, and with a larger value of the LOF, the point would be harder to reach thus it is more likely to be an outlier.

K-nearest neighbors

K-nearest neighbors (KNN) is a distance-based outlier detection method, and the degree of anomaly is determined by the separation between a specific point and its neighbors. There are two popular versions of the anomaly score for the KNN. One is the exact distance to the k-th nearest neighbor, and the other is the average distance for the closest k neighbors. With a higher anomaly score, i.e., a larger the distance to the surrounding neighbors, the higher the chance to be tagged as an outlier for the point considered.

In Table 3.1, all methods mentioned above were summarized in different categories, and a short description was provided for each method. The effectiveness of these methods will be examined using both synthetic and simulated data. The candidate methods identified in the tests will be applied to the real data to assess the validity of the framework.

3.3 Experiments for anomaly detection on flight data records

With the enabler of identifying phases of flight for rotorcraft operations, we can now test the performance of the methods proposed in section 3.2 on flight segments within the same flight phase. Due to no anomalous label attached to our current dataset, synthetic and simulated data with injected anomalies were created to assist in testing the performance of the methods and providing guidance on selecting the methods suitable to the task. This section will first describe the processes of generating synthetic/simulated data and constructing various scenarios. A metric that balances the tradeoff between a false positive and a false negative was chosen for performance evaluation. Once the best methods were identified using synthetic/simulated data, these methods are applied to the flight segments in an actual operation. Results from trajectory pattern mining and time series shape analysis will

Table 3.1: Summary table for methods in different categories

Method	Description
Distance metric	
Euclidean distance	For trajectories in equal length
Hausdorff distance	For trajectories in unequal lengths; not considering the order of waypoints in time axis
Dynamic time warping	For trajectories in unequal lengths; capable of handling asynchronous trajectories
Feature extraction	
FPCA	Using FPC scores to represent the original signals
Nonlinear stacked AE	A nonlinear extension of the traditional principal component analysis
1DCAE	Capable of capturing temporal information of time series
CVAE	The extracted features in latent space follow the assumed probability distribution
Outlier detection	
DBSCAN	A clustering approach supports outlier detection
Isolation forest	Ensemble method
Local outlier factor	Density-based method
KNN	Distance-based method

be presented, and we will include flight segments from different phases, such as initial climb and the approach. As expected, different types of anomalies are detected in the real data using this process, and the situations encountered in real data resemble the scenarios created in the synthetic/simulated dataset.

3.3.1 Using synthetic data for the selection of candidate methods

To test the effectiveness of the methods mentioned in Figure 3.3, we generated synthetic sample data using Gaussian processes, and different scenarios were constructed to account for probable situations we might encounter in the real dataset. The procedure for synthesizing data for a specific pattern starts with a mean signal and then adds noise to the mean to generate sample signals. As shown in Figure 3.6, we have two sets of sample signals which belong to different smoothness settings. Three distinct patterns within each smooth-

ness setting can be mixed and matched for scenario creation. To construct scenarios with injected anomalies, signals from different groups were drawn from this synthetic data pool. Two scenarios are considered in the test:

(1) synthetic data with one dominant pattern and two outlier patterns

(2) synthetic data with two dominant patterns and one outlier pattern

The outlier or anomaly percentage was set at 5% due to its rarity by definition. An example of the cases in the first scenario under the smooth setting is illustrated in Figure 3.7. In this example, the normal signals are composed of all samples from one of the patterns in the synthetic pool, and the anomalous signals are randomly selected from the remaining patterns. Another example of the cases in the second scenario under the rough setting is demonstrated in Figure 3.8. In this example, the normal signals contain all samples from any two patterns in the synthetic pool, and the anomalous signals are drawn randomly from the remaining pattern. The methods in the portfolio will be tested using two scenarios with different smoothness settings, and their performances will be reported on a scenario basis.

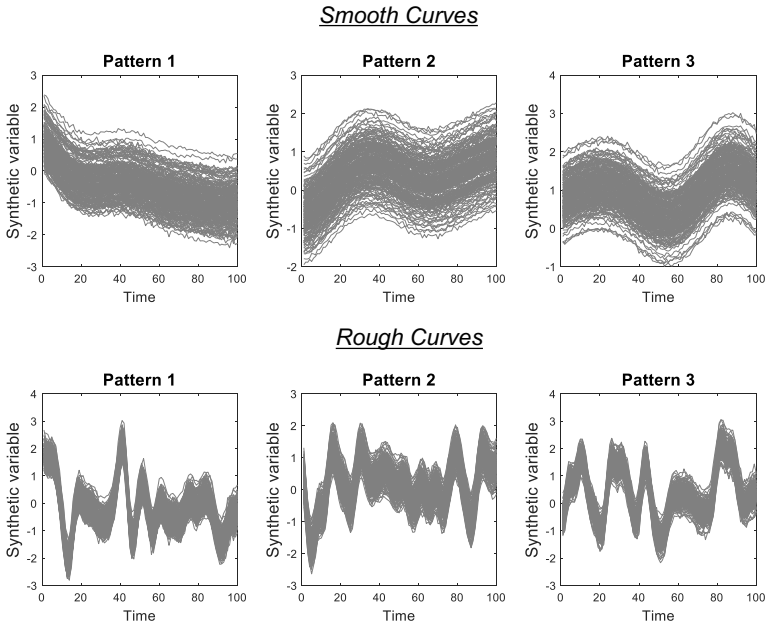


Figure 3.6: Synthetic data pool

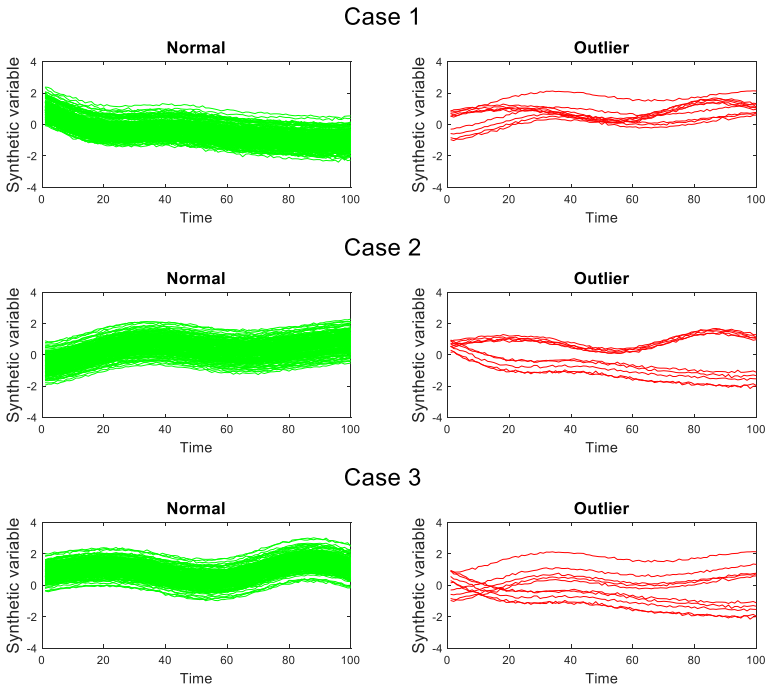


Figure 3.7: Scenario 1: 1 dominant pattern with 2 outlier patterns (smooth setting)

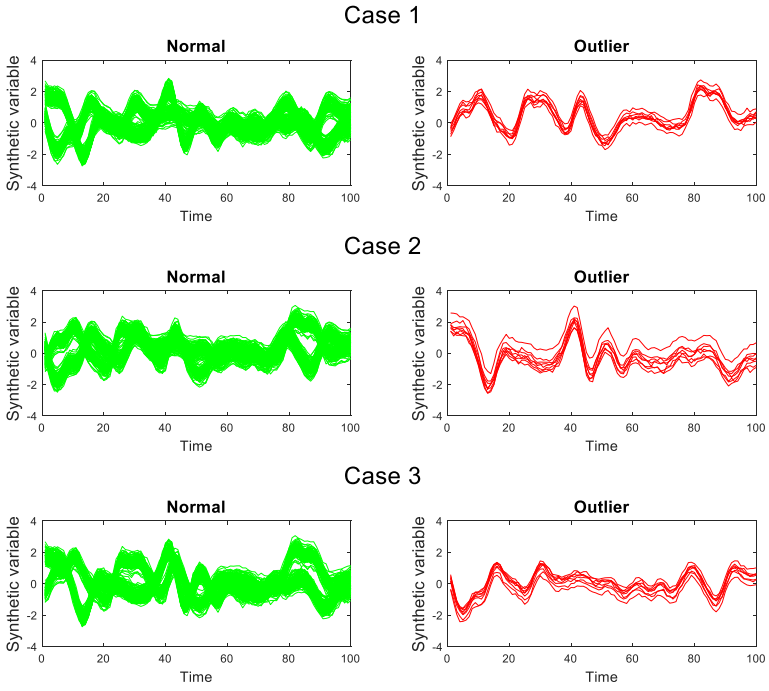
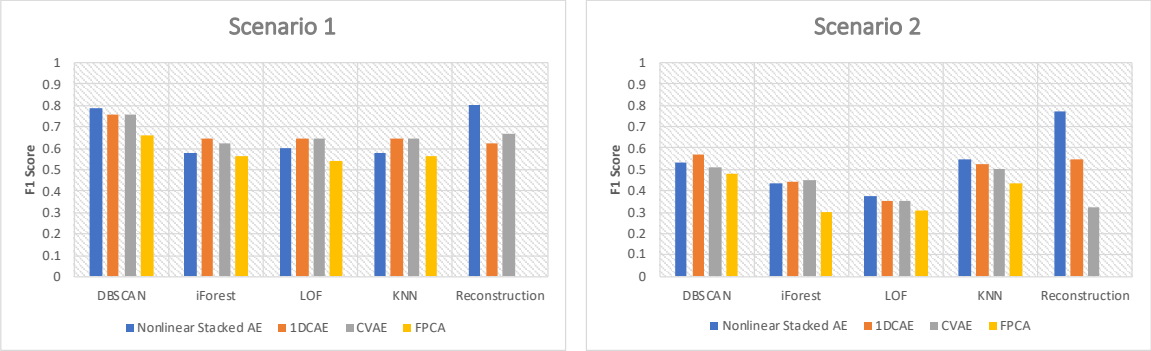


Figure 3.8: Scenario 2: 2 dominant patterns with 1 outlier pattern (rough setting)

For the task of anomaly detection, using accuracy as the metric for evaluating the performance of an algorithm is biased owing to the imbalanced data. If we have an algorithm that consistently does not predict anomalies in the dataset, it can achieve exceptionally high accuracy under the assumption of an extremely low percentage of outliers. Instead of using accuracy, the precision, which stands for the percentage of true anomalies out of all predicted anomalies, and the recall, which denotes the percentage of anomalies being caught out of all true anomalies, are more appropriate for the task. To combine the benefits from the precision and recall, the F1 score, which is the harmonic mean of these two metrics, is adopted as the performance measure in this study.

The comparison of the F1 scores for methods in the portfolio is summarized in Figure 3.9. These methods generally show better performance in the first scenario regardless of the smoothness setting. If there is only one pattern for the normal data, the anomalies are easier to identify than the scenario having multiple normal patterns. When considering the reconstruction error as the anomaly score, the nonlinear stacked autoencoder outperformed other types of autoencoders. For other outlier detection methods, DBSCAN and k-nearest neighbors achieve higher F1 scores overall across different scenarios no matter what feature extraction methods were used. Therefore, the candidate methods selected from this test using synthetic data are all the feature extraction methods in the portfolio along with DBSCAN and k-nearest neighbors, and the nonlinear stacked autoencoder with the reconstruction error as the outlier indicator. These candidates will be applied to a set of simulated data for picking the ideal methods for the entire framework.

Smooth Curves



Rough Curves

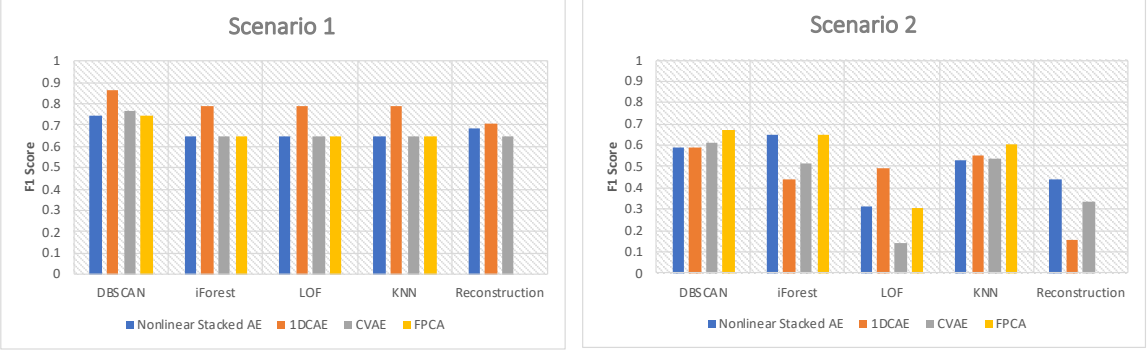


Figure 3.9: F1 scores comparison for methods in the portfolio using synthetic data

3.3.2 Using simulated data to test the candidate methods

To further investigate the candidate methods identified from the test using synthetic data, a trial to simulate normal and abnormal initial climb segments was conducted using an S76-D simulator. The initial climb segment is the first climb segment right after the takeoff. Based on the results from phases of flight identification, the starting altitude of these climb segments may vary from 100 to 200 feet above ground level (AGL). In order to align these segments, an altitude of 100 feet AGL, which is used as the standard threshold for differentiating the low/high altitude maneuvers, is chosen as the starting altitude for the initial climb segments.

From a discussion with the subject matter experts involved in this study, the speeds

for the best angle of climb and the best rate of climb are 55 and 70 knots, respectively. Therefore, we assumed that the normal airspeed for the initial climb segments should vary within the range of 55-90 knots. If the airspeed of a segment falls outside of the specified range most of the time, the segment would be considered an abnormal one. In the trial, 14 runs of normal/abnormal initial climb segments were simulated, and these samples are displayed in Figure 3.10. To increase the sample size in the normal/abnormal groups, we applied Gaussian process regression to generate samples from the labeled simulated segments. The outlier percentage is set at 5%, which is the same as the test for synthetic data. In this test, three scenarios are considered and depicted in Figure 3.11:

(1) one dominant normal pattern with multiple abnormal patterns

(2) multiple dominant patterns with one abnormal pattern

(3) multiple normal and abnormal patterns

The performance of the candidate methods on the simulated initial climb segments is evaluated and presented using an F1 score, and a bar chart that summarizes the results is shown in Figure 3.12. In all scenarios, two methods outperform the rest in the candidate methods: the convolutional variational autoencoder (CVAE) with DBSCAN and functional principal component analysis (FPCA) with DBSCAN. We will apply these two methods along with a trajectory analysis to detect anomalies on the real initial climb and approach segments. The results will be qualitatively evaluated and presented in the following subsection. Through a test with the actual flight segments, the validity of this framework for detecting potential anomalies can be further substantiated.

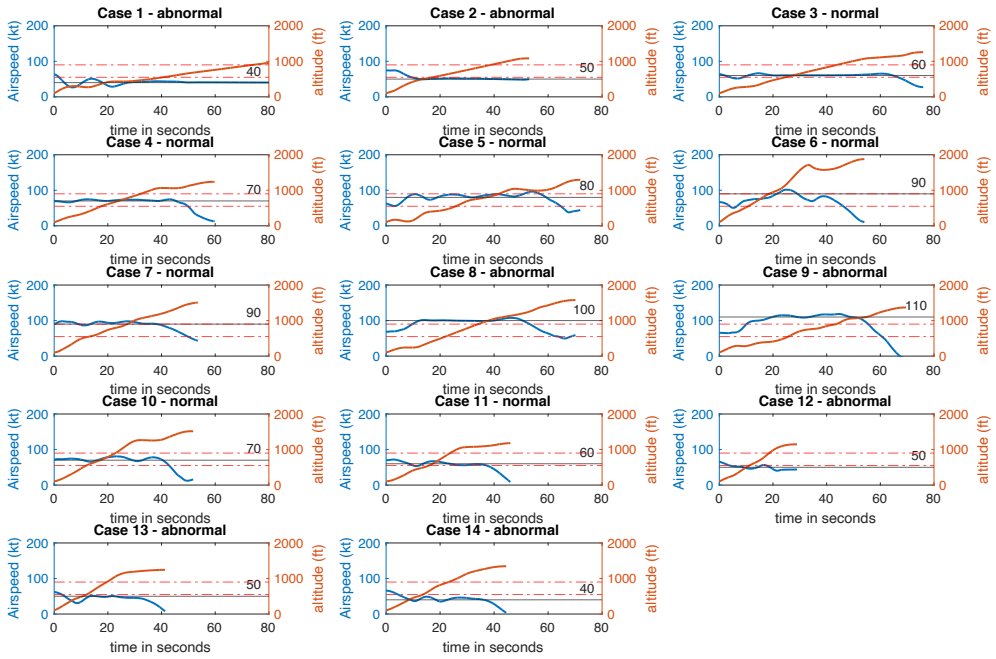


Figure 3.10: Samples of normal and abnormal initial climb segments from a simulated trial

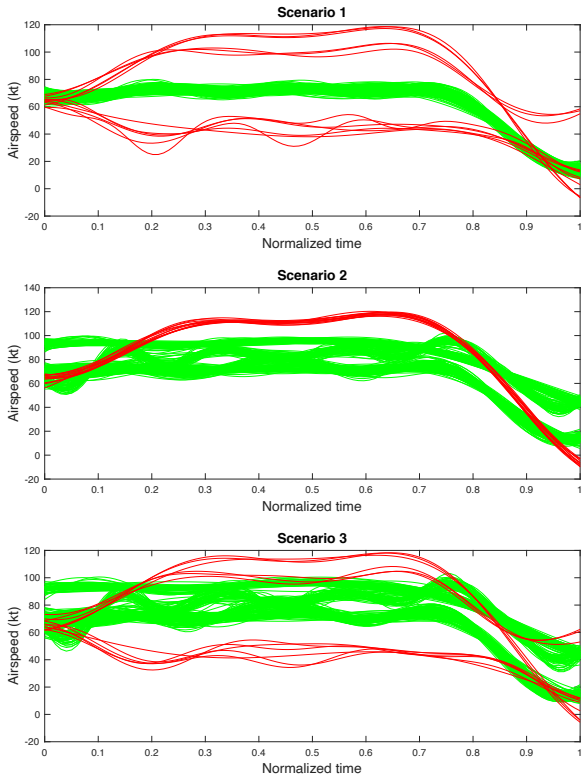


Figure 3.11: Different scenarios considered in the test for simulated segments

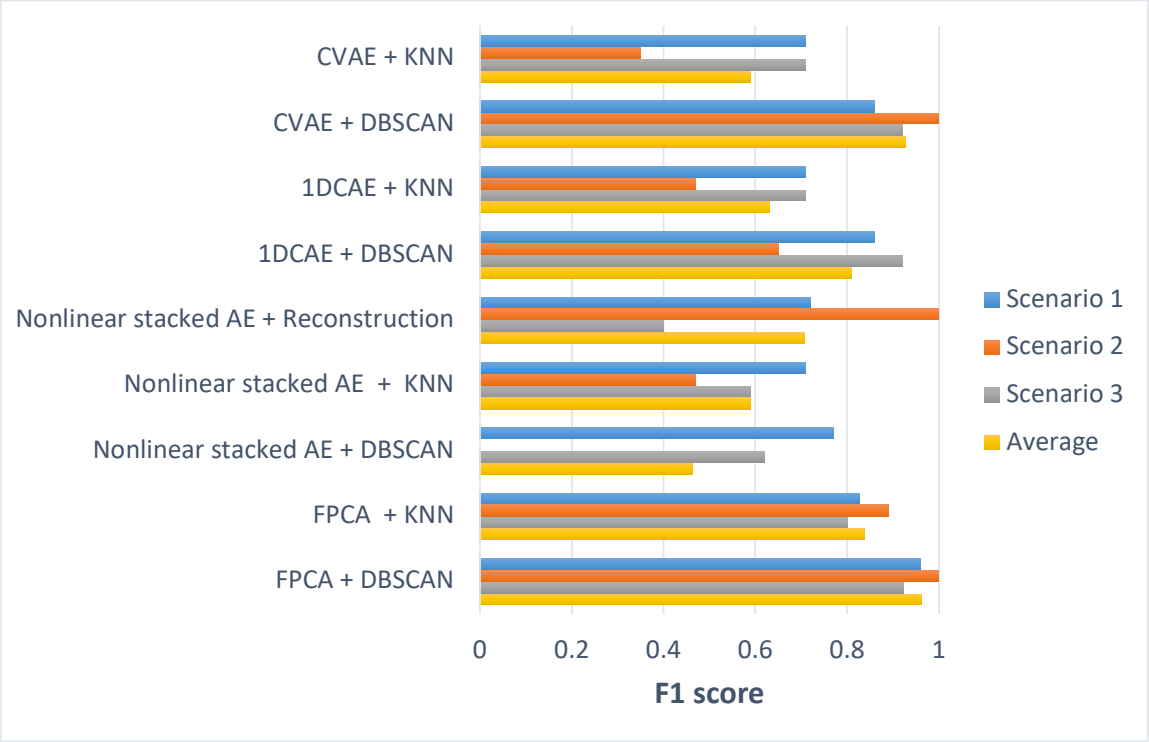


Figure 3.12: F1 scores comparison for the candidate methods using simulated data

3.3.3 Results of anomaly detection on initial climb and the approach segments

We have selected two combinations of methods to detect shape anomalies in signals based on the tests on synthetic/simulated data. Here the flight data from an actual operation will be used to validate the methodology. The test data come from distinct flight segments belonging to the initial climb and the approach. To recap, the proposed framework of anomaly detection on flight data records consists of three key modules: (1) trajectory pattern mining, (2) time-series length analysis, and (3) shape analysis on the flight parameters. These flight segments are obtained from an air ambulance operation, and the segments in the first set correspond to the flight phase of the initial climb. These initial climb segments are prefiltered such that they link up with a specific takeoff site. The number of valid segments for this group is 81, and their trajectories in three-dimensional space are shown in Figure 3.13. The blue and green circles in the figure represent the starting and ending points of

the initial climb segments, respectively. From the results of trajectory pattern mining, all the distance measures discussed in section 3.2.1 are feasible to recognize distinct trajectory patterns. However, we found that Euclidean distance can provide a better and interpretable grouping of the segments. In Figure 3.14, these segments are separated into five different groups based on their corresponding trajectory patterns. We can interpret the logic behind this grouping as trajectory differentiation through the heading direction and the path length. For example, we can observe that each group in the figure has its own heading direction, like group 4 headed toward the southwest while group 5 went to the southeast. Moreover, those with longer path lengths like groups 1 and 3 were placed into their own group as singletons. Based on our definition of anomaly, the groups with the majority of segments should be labeled as normal. The ones with the least number of segments would be considered abnormal in the context of trajectory patterns. In this particular example, the segments in group 4 are regarded as normal and would be brought into the following analyses.

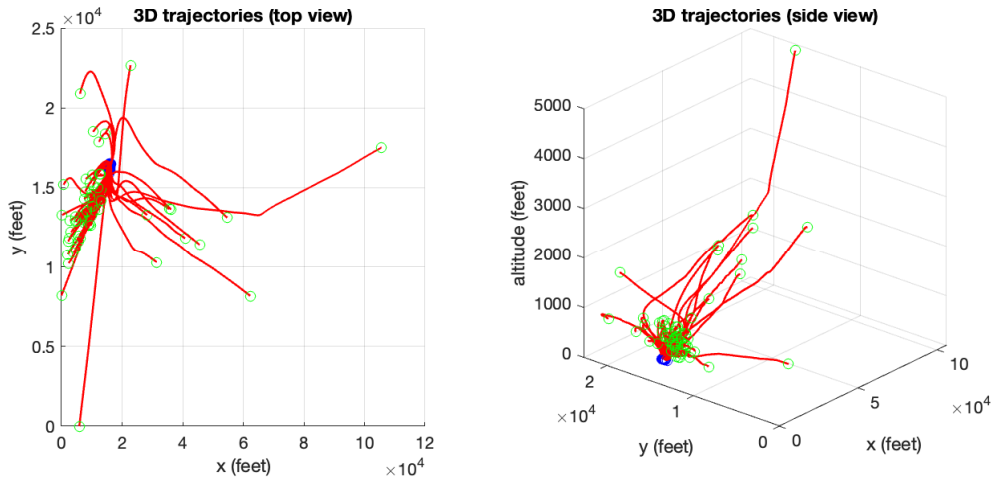


Figure 3.13: Visualization of trajectory data for initial climb segments

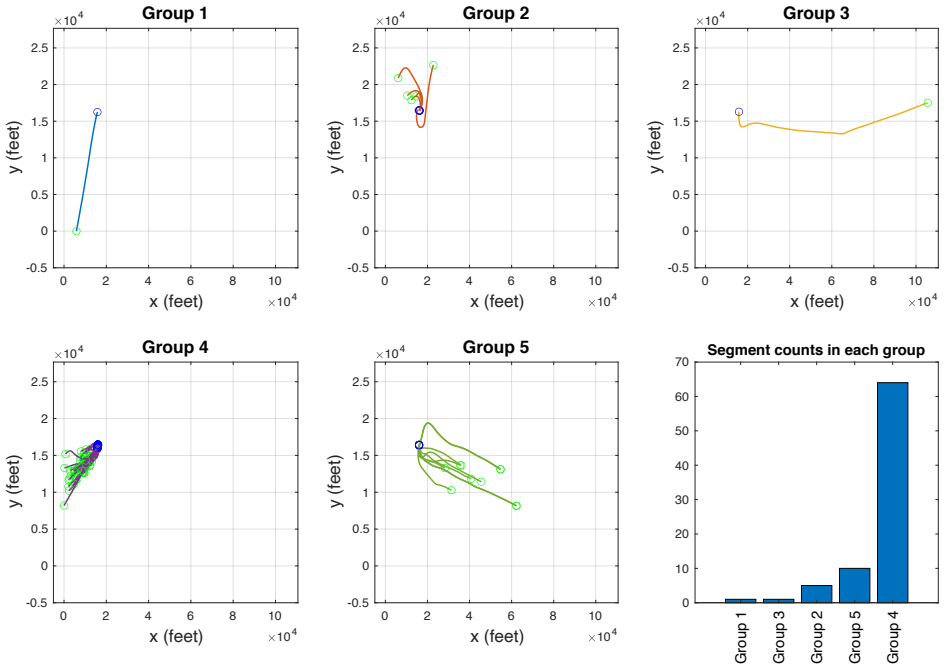


Figure 3.14: Sample results of trajectory pattern mining on initial climb segments

Following the trajectory pattern mining, the segments tagged as normal would be examined through the time series length analysis for filtering out the segments with time duration exceeding the range. Once the segments were within comparable length, the shape analysis was conducted to detect shape anomalies for the flight parameters of interest. Here we chose the ground speed as the targeted flight parameter, and the results from FPCA with DBSCAN and CVAE with DBSCAN are shown in Figure 3.15 and 3.16 separately. The plots displayed on the left of both figures correspond to the shape features extracted using FPCA / CVAE, and the results from DBSCAN outlier detection are displayed using color-coded dots. In the legend from each subplot, the minus one means the data point is an outlier, while the other positive integers stand for the data points in different normal groups. The right subplots provide the exact traces of the flight parameter in each category. Despite having different representations in the feature space, these two methods detect similar flight segments as shape anomalies. Compared with the increasing trend observed in ground speed signals from the normal segments, the anomalous segments have

a profile with a relatively flat beginning followed by a logarithmic growth. From the results of these two methods, the scenario encountered for this particular group of real flight segments corresponds to the scenario with multiple normal patterns and a single abnormal pattern, which was investigated in the tests for synthetic/simulated data. These two methods successfully indicate the capability for detecting shape anomalies, and since they are data-driven approaches, it is anticipated to see the growth in performance with the advent of a larger sample size.

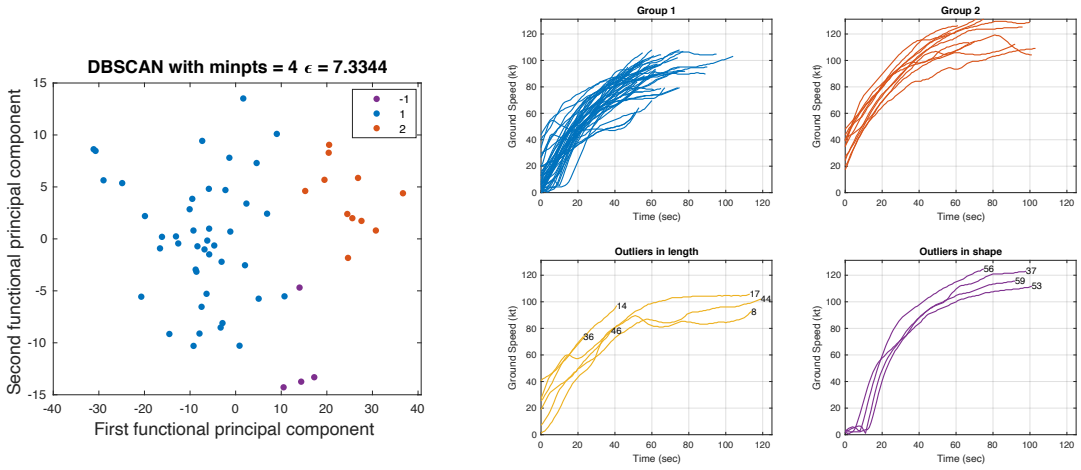


Figure 3.15: A sample result of the shape analysis on initial climb segments using FPCA with DBSCAN

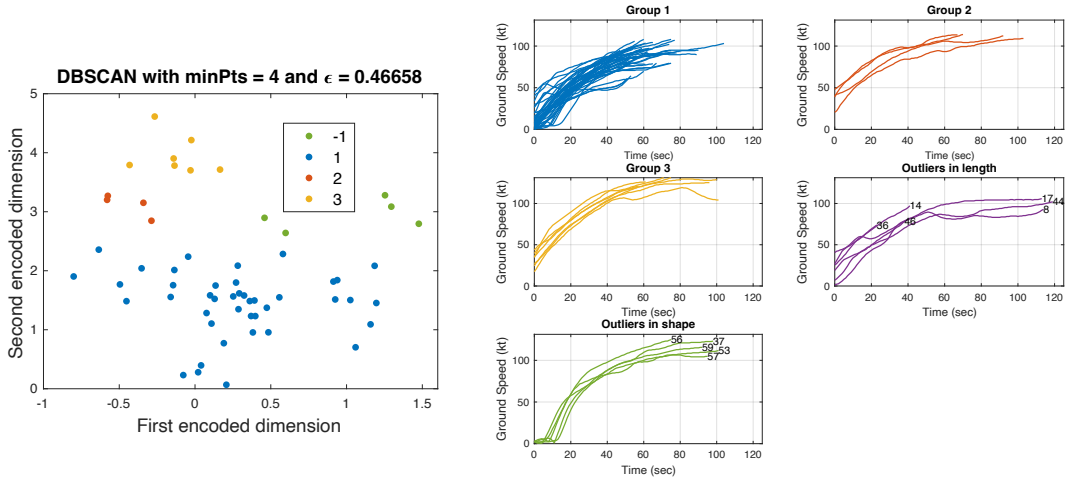


Figure 3.16: A sample result of the shape analysis on initial climb segments using CVAE with DBSCAN

Aside from analyzing the initial climb segments, this anomaly detection framework can also be applied to other segments in different flight phases. The algorithm developed by Robinson et al. [72] was used to extract the approach segments from an air ambulance operation for a specific landing site. The number of valid segments for this group is 331, and the three-dimensional trajectories of these segments are shown in Figure 3.17. Now the blue circles become the starting points of the approaches, and the green circles are the terminal points of these segments. After the trajectory analysis, ten different patterns are found, and they are visualized in Figure 3.18. Here we chose to visualize these trajectories in three-dimensional plots because, for some cases, like groups 3 and 5, their patterns cannot be easily distinguished in a two-dimensional projection. Again, we select the group with the largest number of segments, i.e., group 9, for the following analyses. The flight segments in groups 2 and 4 can be considered outliers in their trajectory pattern. The ground speed is chosen again as the signal of interest to conform with the previous analysis on the initial climb segments. The outliers detected using both FPCA + DBSCAN and CVAE + DBSCAN are reported in Figure 3.19 and 3.20. The flight segments colored in blue from both figures are tagged as normal, but we found that there is one trace situated above all

others, which might be an anomalous segment being undetected. The traces colored in orange are the outliers in shape, and we found that similar flight segments are detected regardless of which method we use. It is observed that the flight segments in the group of outliers in shape do have different patterns compared to the norm defined by the flight segments in group 1.

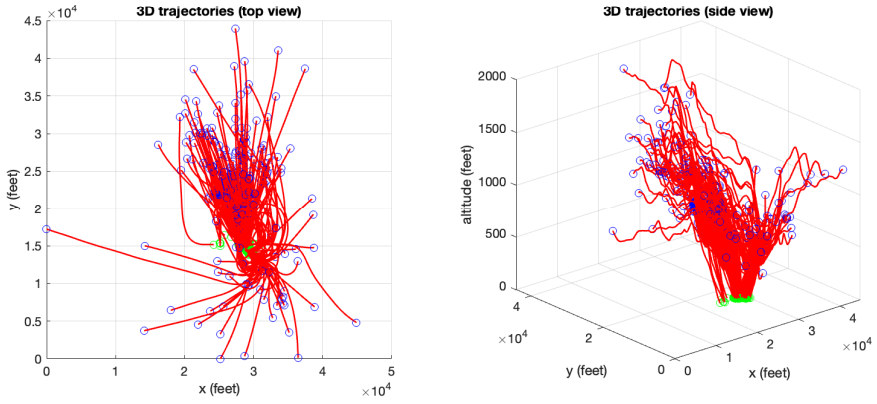


Figure 3.17: Visualization of trajectory data for the approach segments

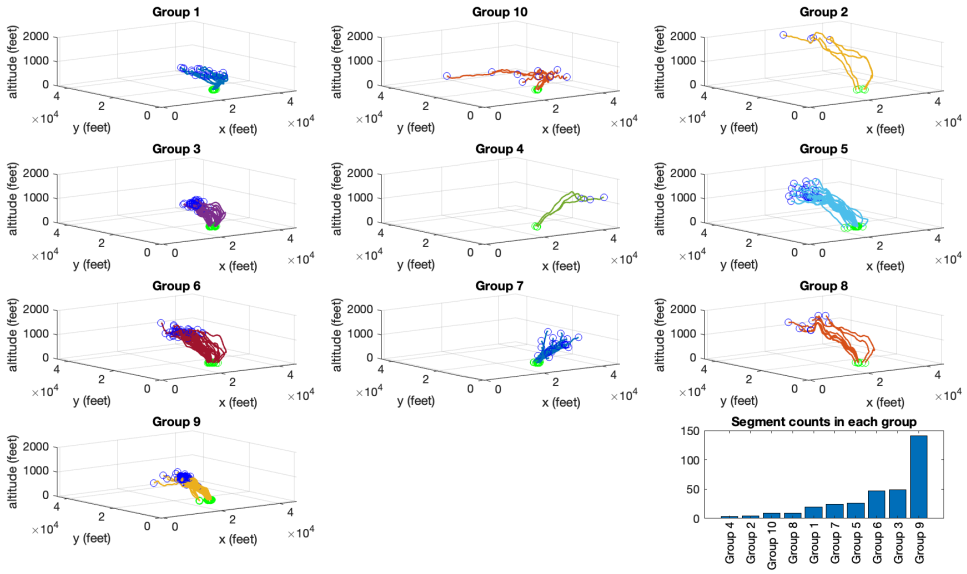


Figure 3.18: Sample result of trajectory pattern mining on the approach segments

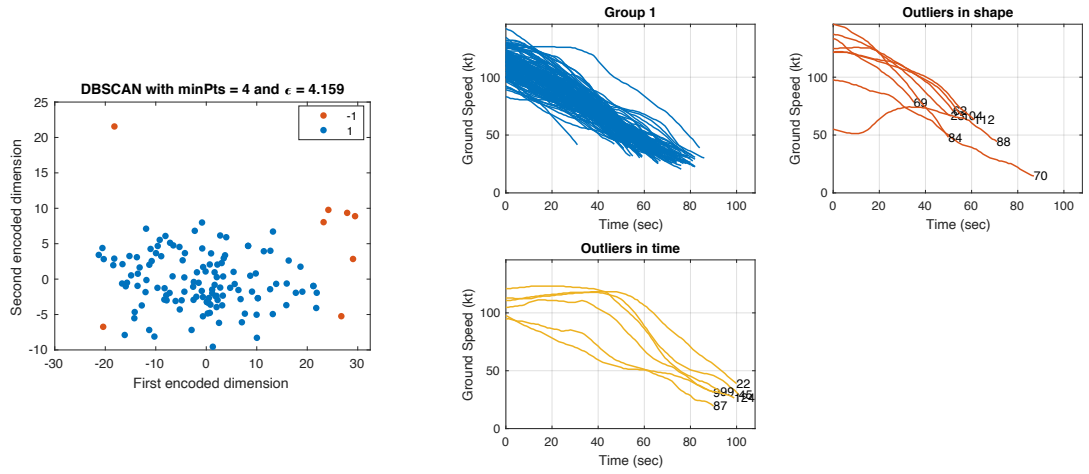


Figure 3.19: A sample result of the shape analysis on the approach segments using FPCA with DBSCAN

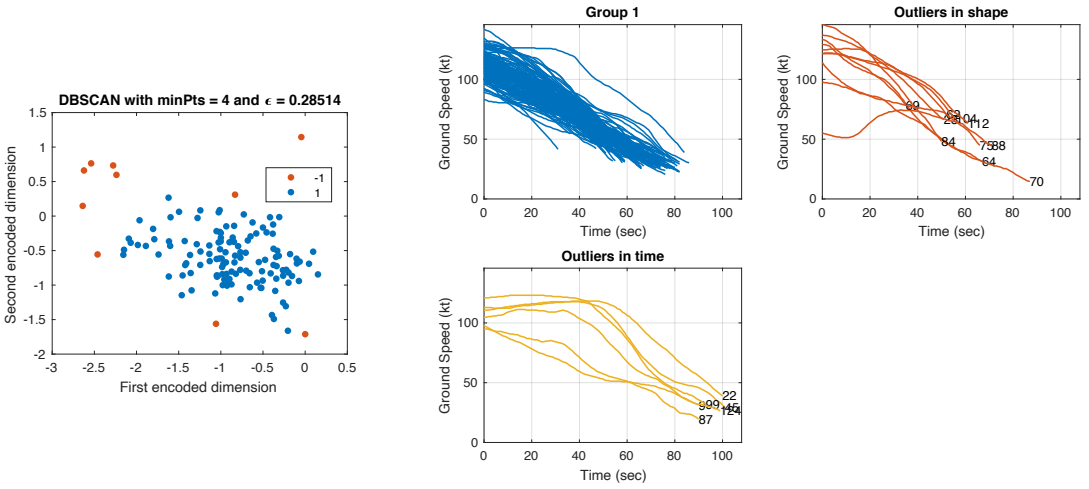


Figure 3.20: A sample result of the shape analysis on the approach segments using CVAE with DBSCAN

3.4 Summary for the third research question

We proposed a sequential approach that contains elements including trajectory pattern mining, time series length analysis, and shape analysis to detect the collective anomalies from flight data records for rotorcraft operations. Due to the inaccessibility of a labeled dataset,

synthetic and simulated data were created to evaluate the performance of methods in the candidate list. For trajectory pattern mining, using Euclidean distance as the distance measure along with hierarchical clustering for pattern grouping is a viable approach to place similar trajectories into the same group. To detect shape anomalies for time series within comparable length, we found that both functional principal component analysis (FPCA) and convolutional variational autoencoder (CVAE) are useful for capturing the shape information with features in relatively low dimensions. By applying the density-based clustering DBSCAN on these features, the flight segments not following the majority pattern in the group are detected. Compared to traditional exceedance analyses, the proposed methodology can detect potential anomalies without specifying thresholds on flight parameters.

CHAPTER 4
EXPLORATORY ANALYSIS ON THE AUTOROTATION

In the previous two chapters, a retrospective analysis that can assist in finding potential anomalies in flight data records is explained. To move a step forward, a more proactive approach, which corresponds to the exploratory analysis, aims to avoid hazardous events and know how to react once entering those events. We picked the autorotation as our hazardous event because it meets the criteria of time criticality and low occurrences in operation. However, the methodology does not tailor specifically to the autorotation, and it should work on other hazardous events as well. In this chapter, we will first formulate the research questions regarding the exploratory analysis along with the corresponding hypotheses and experiments. Then, a roadmap of the analysis will be provided. The remaining sections will explain the main elements in the roadmap and perform the experiments to test the hypotheses stated in the problem formulation. In the end, we will summarize the findings from the exploratory analysis on the autorotation.

4.1 Problem formulation

The second research objective in this study is to develop a methodology for efficiently exploring the safety envelope and retrieving the action for recovery in a hazardous event. The autorotation maneuver is selected as the use case for testing and validating the methodology. To explore the envelope for the autorotation, a mathematical representation capable of capturing the physics of the maneuver needs to be developed. Models with different fidelity ranging from a simple two-dimensional point-mass to a more sophisticated three-dimensional rigid-body model can be built based upon the task's requirement. Sometimes, the selection of model fidelity is a compromise between the computation time and the accuracy. To find the boundary of the safety zone and investigate the feasibility of using a surro-

gate for predicting the required controls, a two-dimensional model in a vertical plane would be sufficient, and this model has been widely used in the literature [42][44][47][53][52].

Once the model is constructed, it is required to go through a verification process to make sure it is built correctly. If a human pilot conducts the simulation, multiple paths exist from an initial condition to the end condition, and these trajectories would depend on different levels of piloting skills. In order to have a one-to-one mapping between an initial condition and its corresponding terminal condition, optimal control is a method to generate the desired trajectory objectively. Further, this one-to-one relationship is vital if we intend to apply a surrogate-type of analysis for exploring the operational space.

Due to an optimization process being involved in the exploratory analysis, depending on the complexity of the simulation model and the solver chosen, it might be computationally expensive to get the result from just a couple of scenarios. To tackle this issue, some techniques in the design of experiments (DoE) and surrogate modeling will be investigated in the analysis, and they should considerably increase the efficiency of the exploration process. In general, a space-filling design would be a typical choice for a computer experiment. With the data collected from the chosen experimental design, a surrogate model can be built for predicting the terminal conditions and the controls for recovery. Suppose the surrogate does not perform as expected; in that case, augmented points may be added to the original experimental design for probing more widely on the response surface. Thus, the performance of the surrogate can be improved. Lastly, a sensitivity analysis can be used to investigate the relative importance of the input variables on the response considered. The concept of the methodology described here is summarized in Figure 4.1.

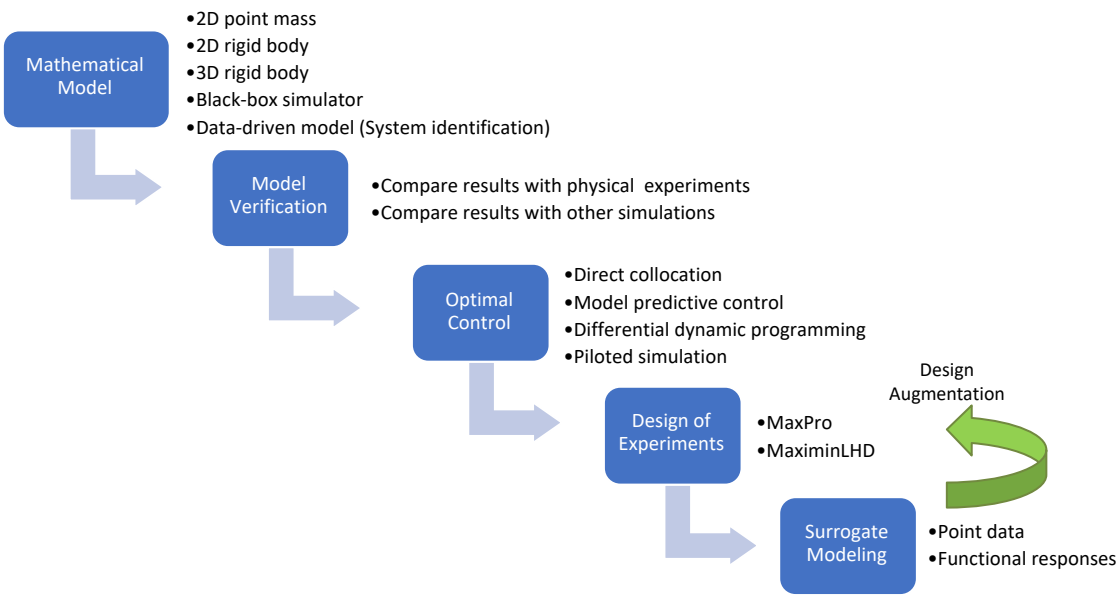


Figure 4.1: An overview of the methodology for the exploratory analysis

Since we will primarily place our focus on surrogate modeling, the first research question posed for the exploratory analysis is stated as follows:

RQ4-A: Among various kinds of surrogate models suitable for computer experiments, which one is appropriate for the task of exploring the operational space?

The corresponding hypothesis is described below:

H4-A: The Gaussian process regression is a flexible method used in computer experiments to create surrogates specifically for expensive-to-evaluate computer codes. It supports uncertainty quantification and can also predict functional responses.

To test the hypothesis, the following experiment will be conducted:

E4-A: Several candidate surrogate models will be tested on some known functions in either low or high dimensional space to evaluate the performance.

Once the surrogate model is selected, the number of design points for the experiment should be determined. Thus, the following research question is asked:

RQ4-B: How many runs are needed or sufficient for the experiment to approximate the actual response of an unknown function accurately?

The hypothesis constructed to address the question is stated as follows:

H4-B: A metric that is useful for approximately inferring the trend of the accuracy is required to provide guidance on the selection of run numbers.

The experiment to test the above hypothesis is briefly described below:

E4-B: Test the effectiveness of the proposed metrics on some known functions used in E4-A.

In the second research objective O2, we intend to use the surrogate for predicting the safety envelope and estimating the optimal controls of the entire region. To achieve the objective, the following research question is posed:

RQ4-C: The computed optimal controls in the designed locations of the operational space are in the form of functional responses, and they are in different lengths. What method is useful for predicting this type of response?

The corresponding hypothesis is formulated as follows:

H4-C: A functional response can be characterized by its length and shape. By employing a surrogate model with two Gaussian processes, the length and shape information of the signal can be captured separately.

The experiment to test the above hypothesis is described below:

E4-C: The proposed surrogate model will be constructed using the data acquired from a set of experimental runs in the height-velocity space. To evaluate its effectiveness, the predictions from this model will be compared with the results from the simulation.

Last, more input variables can be added to the surrogate model beyond initial horizontal speed and initial height. With a more extensive set of input variables, it is essential to know the key variables affecting the response. Thus, the following question is posed:

RQ4-D: Among all variables being investigated in the operational space, which variable plays the most significant role in the response?

The corresponding hypothesis is formulated as follows:

H4-D: A sensitivity analysis based on functional analysis of variance (ANOVA) and Sobol sensitivity indices can be used to identify the critical variables.

To test the above hypothesis, the following experiment will be conducted:

E4-D: Perform an experiment on factors including initial height, initial horizontal velocity, and vehicle weight, and then use the sensitivity analysis to determine the key variables.

4.2 Methods used in the exploratory analysis

In this section, we will describe the methods used in the exploratory analysis followed the order of the sequence in the roadmap shown in Figure 4.1. The helicopter model suitable for investigating the autorotation maneuver is first introduced, and then the approach for finding the optimal control is presented. Further, a logical rationale for the selection of the design of experiments is provided, and several candidate surrogates for computer experi-

ments are put forward for comparison. In the end, we will explain a sensitivity analysis built upon the surrogate model for identifying the key input variables that affect the response.

4.2.1 Helicopter modeling

In our investigation, a two-dimensional helicopter model in a vertical plane for the autorotation developed in [42] was modified and used to test the validity of the overall framework. Due to its simplicity, the following assumptions are made to the model:

- The motion of the vehicle is restricted to a vertical plane
- Dynamic inflow is ignored, and the calculation of induced velocity at vortex ring state is based on an empirical formula
- Rotor compressibility effect and tail rotor power loss are neglected
- Fuselage drag is calculated from an equivalent flat plate area
- The pitch angle of the helicopter is approximated by the tilt of rotor tip-path-plane
- Mean profile drag of the rotor is typically dependent on the angle of attack of the rotor blade, but it is assumed to be a constant here
- Ground effect, which provides cushioning to the vehicle when it is close to the ground, is neglected
- No wind encounter and no air density change during the entire maneuver

The free-body diagram of the two-dimensional model in a vertical plane is shown in Figure 4.2 and the physical meaning of each parameter is documented in Table 4.1 . The parameters of the model are based on a Bell OH-58A helicopter.

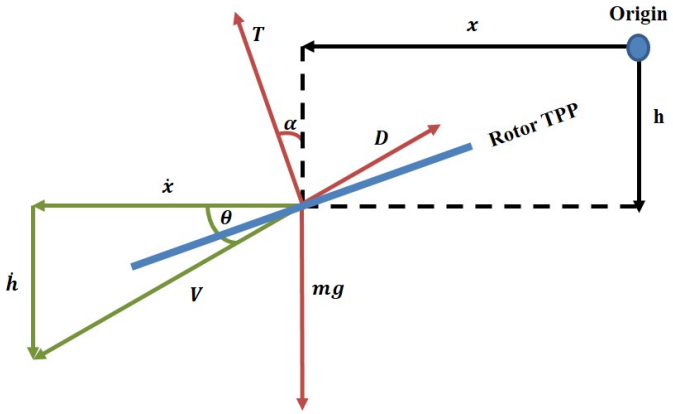


Figure 4.2: Free body diagram

Table 4.1: Parameters used in the two-dimensional model in a vertical plane

Parameter	Meaning	Value
f_e	equivalent flat plate area, ftˆ2	24
ρ	air density, slug/ftˆ3	0.002378
σ	rotor solidity	0.048
R	rotor radius, ft	17.63
mg	weight of helicopter, lb	3000
Ω_0	nominal rotor rpm	352
c_d	mean profile drag coefficient of rotor blades	0.0087
I_R	rotor inertia, slug-ftˆ2	1344
K_{ind}	induced velocity correction factor	1.13

The force balance equations in horizontal and vertical directions are shown below

$$\begin{aligned} m\dot{w} &= mg - T\cos\alpha - D\sin\theta \\ &= mg - \rho(\pi R^2)(\Omega R)^2 C_{Tz} - \frac{1}{2}\rho f_e w\sqrt{u^2+w^2} \end{aligned} \tag{4.1}$$

$$\begin{aligned} m\dot{u} &= T\sin\alpha - D\cos\theta \\ &= \rho(\pi R^2)(\Omega R)^2 C_{Tx} - \frac{1}{2}\rho f_e u\sqrt{u^2+w^2} \end{aligned} \tag{4.2}$$

where T is the thrust, D is the drag, α is the tilt of tip-path-plane, θ is the flight path angle,

u is the horizontal velocity, v is the vertical velocity, C_{Tx} and C_{Tz} are the thrust coefficients in x and z directions respectively.

Torque balance equation is expressed as follows:

$$I_R\dot{\Omega} = -Q_{req} = -[\rho(\pi R^2)(\Omega R)^2 R] \cdot C_Q \tag{4.3}$$

where Q_{req} is the required torque, C_Q is the torque coefficient.

The kinematic relations are stated as follows:

$$\dot{h} = w \tag{4.4}$$

$$\dot{x} = u \tag{4.5}$$

here the direction convention in the vertical axis is taken as downward positive.

The non-dimensional quantities used in the formulation are described as below:

$$\begin{aligned} C_Q \approx C_P &= \frac{1}{8}\sigma c_d + C_T\lambda \\ C_{Tx} &= C_T \sin\alpha \\ C_{Tz} &= C_T \cos\alpha \\ \mu &= \frac{u\cos\alpha + w\sin\alpha}{\Omega R} \\ \lambda &= \frac{u\sin\alpha - w\cos\alpha + \nu}{\Omega R} \\ \nu &= K_{ind}\nu_h f_I f_G \end{aligned} \tag{4.6}$$

where C_P is the power coefficient, C_T is the thrust coefficient, λ is the rotor inflow ratio, μ is the advanced ratio, ν is the induced velocity, ν_h is the hover induced velocity, f_I is the induced velocity parameter, and f_G is the ground effect on induced velocity. It is set to 1 for ignoring the ground effect. The induced velocity parameter f_I is calculated using the

following formula, which accounts for the vortex ring state (VRS) condition:

$$
\begin{aligned}
f_I &= \begin{cases} 1/\sqrt{(b^2+(a+f_I)^2)}, & \text{if } (2a+3)^2+b^2 \leq 1.0. \\ a(0.373a^2+0.598b^2-1.991), & \text{otherwise.} \end{cases} \\
a &= \frac{u\sin\alpha - w\cos\alpha}{\nu_h} \\
b &= \frac{u\cos\alpha + w\sin\alpha}{\nu_h}
\end{aligned}
\tag{4.7}
$$

In order to facilitate the optimization process, the dynamical equations (Equation 4.1 to 4.3) and kinematic relations (Equation 4.4 and 4.5) are standardized such that the variables are all in the same scale. The resulting equations are shown as follows:

$$
\begin{aligned}
\dot{x_1} &= g_0 - m_0(u_1 x_3^2 + \bar{f} x_1 \sqrt{x_1^2+x_2^2}) \\
\dot{x_2} &= m_0(u_2 x_3^2 - \bar{f} x_2 \sqrt{x_1^2+x_2^2}) \\
\dot{x_3} &= -i_0 x_3^2 (c_0 + \lambda\sqrt{u_1^2+u_2^2}) \\
\dot{x_4} &= 0.1 x_1 \\
\dot{x_5} &= 0.1 x_2
\end{aligned}
\tag{4.8}
$$

where normalized states and controls are

$$
\begin{aligned}
x_1 &= \frac{w}{0.01\Omega_0 R} \\
x_2 &= \frac{u}{0.01\Omega_0 R} \\
x_3 &= \frac{\Omega}{\Omega_0} \\
x_4 &= \frac{h}{10R} \\
x_5 &= \frac{x}{10R} \\
u_1 &= 10^3 C_{Tz} \\
u_2 &= 10^3 C_{Tx}
\end{aligned}
$$

Two modifications are made to the original formulation. First, the controls are changed from C_{Tx} and C_{Tz} to C_T and α because only the constraint on C_T is known. Second, without restricting the change rate in controls, it is likely to have highly fluctuating control signals for some instances. By adding the rate control, additional variables are introduced, and the resulting formulation is shown as below:

$$\begin{aligned}
\dot{x_1} &= g_0 - m_0(x_6 \cos x_7 x_3^2 + \bar{f} x_1 \sqrt{x_1^2 + x_2^2}) \\
\dot{x_2} &= m_0(x_6 \sin x_7 x_3^2 - \bar{f} x_2 \sqrt{x_1^2 + x_2^2}) \\
\dot{x_3} &= -i_0 x_3^2 (c_0 + \lambda x_6) \\
\dot{x_4} &= 0.1 x_1 \\
\dot{x_5} &= 0.1 x_2 \\
\dot{x_6} &= u1 \\
\dot{x_7} &= u2
\end{aligned} \tag{4.9}$$

with the new control variables of

$$\begin{aligned}
u_1 &= 10^3 \dot{C_T} \\
u_2 &= \dot{\alpha}
\end{aligned}$$

4.2.2 Optimal control

The general formulation of an optimal control problem (OCP) is stated as follows:

$$\begin{aligned}
\textbf{Objective functional}: &\quad J = \phi(t_f, \boldsymbol{x}_f) + \int_{t_0}^{t_f} g(t, \boldsymbol{x}, \boldsymbol{u}) dt \\
\textbf{System Dynamics}: &\quad \dot{\boldsymbol{x}} = \mathbf{f}(t, \boldsymbol{x}, \boldsymbol{u}) \\
\textbf{Constraints}: &\quad \mathbf{c}_{min} \leq \mathbf{c}(t, \boldsymbol{x}, \boldsymbol{u}) \leq \mathbf{c}_{max} \\
\textbf{Boundary conditions}: &\quad \mathbf{b}_{min} \leq \mathbf{b}(t_0, \boldsymbol{x}_0, t_f, \boldsymbol{x}_f) \leq \mathbf{b}_{max}
\end{aligned} \tag{4.10}$$

where $\boldsymbol{x}$ is a vector of state variables, $\boldsymbol{u}$ is a vector of control variables, subscript 0 and f represent initial and final conditions, respectively. Generally, it can be thought of as an optimization problem in the functional space where the computed trajectories need to satisfy the system dynamics, constraints, and boundary conditions. The problems with a linear system and a quadratic loss function have a closed-form analytic solution, and it is called the linear quadratic regulator (LQR) problem. However, most of the dynamical systems are nonlinear in nature. Thus, we often need to rely on numerical approaches to solve optimal control problems.

There are two schools of thought for solving an optimal control problem mentioned above. One is to use an indirect method, and it is a traditional way of obtaining optimal control through Pontryagin's maximum principle. It got its name of "indirect" because the state and control variables are not solved directly in the formulation. Instead, the solutions are dependent on the necessary and sufficient conditions for optimality. For a simple problem, constructing the adjoint equations and the Hamiltonian function may be easy. Still, the task becomes difficult to tackle as the situation gets more complex with path inequalities. Also, there are several issues mentioned in [46] with indirect methods, such as the calculation of derivatives of the Hamiltonian and the solution robustness. Therefore, direct methods that specifically minimize the objective function through a transcription process have recently gained more popularity. In direct methods, the optimal control problem is transformed into a nonlinear programming problem (NLP) through discretization. It is easier to solve a parameter optimization than the optimization in functional space. Although there exist other approaches, such as model predictive control (MPC) and differential dynamic programming (DDP) for solving an optimal control problem, in this study, we will adopt the direct collocation method with trapezoidal quadrature to compute the optimal control trajectories. The direct collocation method is briefly explained in the following paragraph.

Assume the time duration of the maneuver from t_0 to t_f can be discretized into a grid

with N elements where

$$0 = t_1 > t_2 > \cdots > t_{N-1} > t_N = t_f \quad \text{and} \quad t_{k+1} - t_k = h_k \quad \text{for} \quad k = 1 \cdots N$$

The corresponding state and control variables in a grid representation are

$$x_1^i \cdots x_k^i \cdots x_N^i$$
$$u_1^j \cdots u_k^j \cdots u_N^j$$

where the superscript i and j are the numbers of dimensions in state and control variables. With the discretization, the original optimal control problem in Equation 4.10 can be transformed into the following form with trapezoidal quadrature for integral:

$$\begin{aligned}
\textbf{Objective function}: \quad & \bar{J} = \phi(t_f, \boldsymbol{x}_f) + \sum_{k=0}^{N-1} \frac{h_k}{2}(g_k + g_{k+1}) \\
\textbf{System Dynamics}: \quad & x_{k+1} = x_k + \frac{h_k}{2}(f_{k+1} + f_k), \quad k = 1, ..., N-1 \\
\textbf{Path and Boundary Constraints}: \quad & \boldsymbol{x}_{min} \leq \boldsymbol{x}_k \leq \boldsymbol{x}_{max}, \qquad \boldsymbol{u}_{min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{max}
\end{aligned} \tag{4.11}$$

If we collocate all variables together as a vector displayed as below:

$$\boldsymbol{z} = [x_1^1, \cdots, x_N^1, \cdots, x_N^I, u_1^1, \cdots, u_N^1, \cdots, u_N^J, t_f]$$

The problem can be formulated in a general form of NLP:

$$\min \bar{J}(\boldsymbol{z}) \tag{4.12}$$

subject to

$$\mathbf{L} \leq \begin{Bmatrix} \boldsymbol{z} \\ A\boldsymbol{z} \\ \mathbf{C}(\boldsymbol{z}) \end{Bmatrix} \leq \mathbf{U} \tag{4.13}$$

where A is the coefficient matrix of linear constraints, $\mathbf{C}$ is the nonlinear constraint function, $\mathbf{L}$ and $\mathbf{U}$ are the lower and upper bounds of the constraints, respectively. The standard objective function for the autorotation problem is to minimize the speed at the touchdown point, and this objective function only contains the terminal cost term. To encourage the convergence in the solution, a running cost that penalizes the deviation from the initial rotor speed is added to our formulation. The overall objective function is shown below:

$$J = u(t_f)^2 + w(t_f)^2 + \int_{t_0}^{t_f} (\Omega(t) - \Omega(t_0))dt \tag{4.14}$$

An open-source solver based on direct collocation method called OptimTraj [80] is used to assist the task of finding the optimal trajectories. Other commercial software like GPOPS-II® can achieve the task as well.

4.2.3 Design of experiments

Design of experiments (DOE) is a field of study dealing with how to place experimental points in the design space meticulously and parsimoniously to gather most of the information from an actual function. Combined with a surrogate model, it is a useful tool for exploring the response of a system, especially when dealing with a computationally expensive model. In this study, the application of the design of experiments has been extended from designing a novel system to finding the safety envelope for a hazardous event of a dynamical system. In a hazardous event, the vehicle would start from a hazardous state, and depending on the control trajectory exerted along the path, it may or may not end in a safe state. For example, a tire blowout instance for a car driven on a highway can be con-

sidered as a hazardous state. Based on various factors, such as the states of the vehicle at the instance of the blowout, and the control strategy selected for steering the car, it may or may not stop successfully on the curbside without crashing with obstacles. Since multiple paths exist from the start to the end of the process, it is required to establish a one-to-one mapping from the initial condition to the terminal condition. The aforementioned optimal control method would be useful for achieving this mapping. However, the optimal path calculation may be time-consuming due to the optimization process on a high-fidelity dynamical model. Through leveraging surrogate modeling with the design of experiments for constructing a safety envelope, it can certainly reduce the time for the exploration process.

Traditional methods of design of experiments originated from agricultural and industrial experimentation, and they were developed primarily for physical experiments. With the advancement in computing technology, more investigations have been moved from physical to virtual ones, which would rely on computer models. There are several differences between physical and computer experiments. First, no noise would be observed in the response for a deterministic computer model. Therefore, replications of a design point used in the physical experiment are not required for computer experiments. Second, hidden variables do not exist in a computer experiment compared with a physical experiment. Thus, methods used to alleviate the impact of hidden variables, such as randomization and blocking, are not necessary for a computer experiment. Third, there is no hard-to-change variable in a computer experiment. As a consequence, the design points can be selected at any location within the range for each variable. With all the differences mentioned above, filling the design space evenly with design points for a computer experiment is intuitive. A "space filling" design is a type of design suitable for the need of a computer experiment. Joseph et al. [63] provided a comprehensive review on some popular choices for space-filling designs, and some highlights will be discussed for guiding our selection of the design of experiments.

Latin hypercube design (LHD) is a type of space-filling design described by Mckay et

al. [81], and it is constructed through random permutations of evenly spaced points in each coordinate. An example of two different LHDs in a two-dimensional space with four runs is shown in Figure 4.3. It is observed that these two designs both have good coverage in the one-dimensional projection. However, design (A) is superior to design (B) because the entire space is better filled with points in design (A) than design (B). To find a better LHD within all possible LHDs, Tang [82] proposed an Orthogonal Array-based Latin Hypercube Design (OALHD). This design would encourage good coverage in the two-dimensional projection, and an example of an OALHD in a four-dimensional space with nine runs is shown in Figure 4.4. We can see that this design provides suitable space-filling property in the one-dimensional projection and the two-dimensional projection. Nonetheless, this design does not guarantee to fill the space uniformly for design points in higher dimensional spaces.

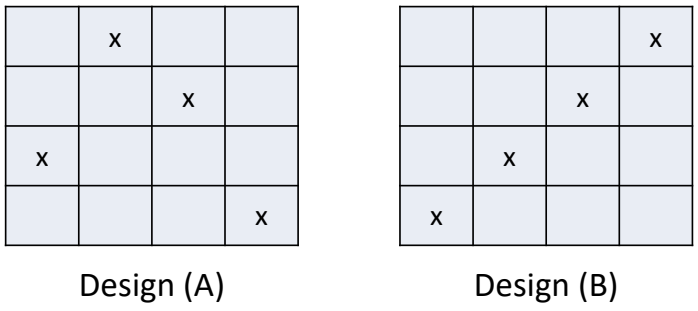


Figure 4.3: Two different Latin Hypercube Designs with four runs in a 2D space

To better fill the space in a high dimensional space, distance measures are used to determine the locations for design points. Johnson et al. [83] introduced two different distance-based designs, the miniMax distance design (mM design) and the Maximin distance design (Mm design). The mM design ensures that all points in the region of interest are close to the design points as much as possible. On the other hand, the Mm design ensures that the design points are distributed far away from each other and prevents local agglomerations of design points. These two distance-based designs provide good coverage in the full-dimensional space but may not retain this property in the lower dimensional pro-

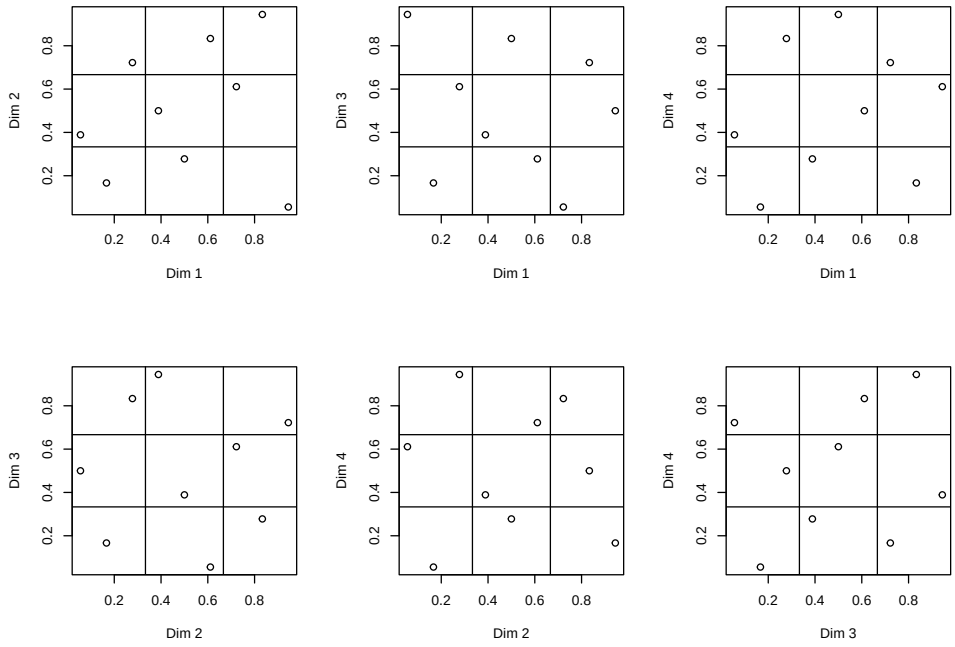


Figure 4.4: An example of Orthogonal Array-based Latin Hypercube Design

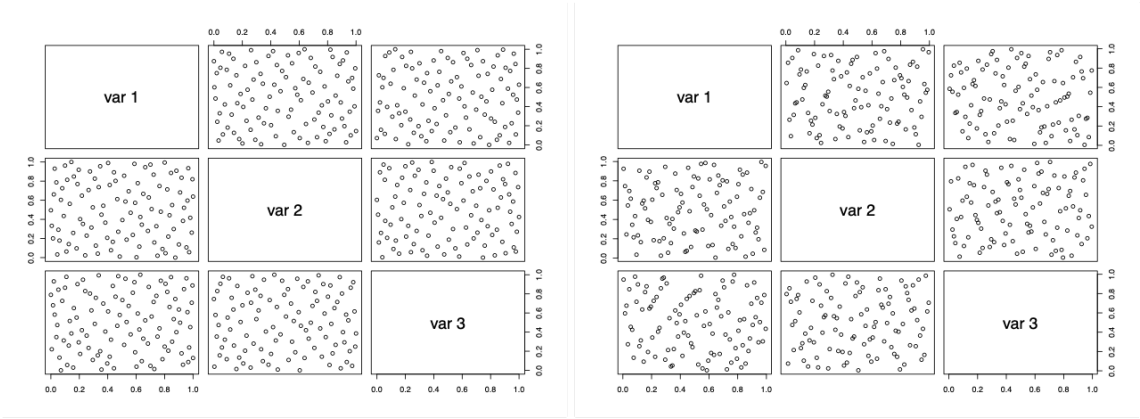
jections. To account for this issue, Morris et al. [84] combined the advantages of LHD and Mm distance metric and developed the Maximin Latin Hypercube Design (MmLHD). This design ensures good coverages in both the full-dimensional space and the one-dimensional projection, but it does not guarantee having good coverages in the remaining subspaces.

Joseph et al. [62] aimed to resolve the coverage issue in the subspace spanned by factors and proposed Maximum Projection design (MaxPro). This design modified the MmLHD criterion with the weighted Euclidean distance, and it adopted the Bayesian framework for constructing the MaxPro criterion. If the weights on Euclidean distance are assumed to follow a uniform distribution, then the criterion can be expressed as follows:

$$\min_{\mathbf{D}} \sum_{i}^{n-1} \sum_{j=i+1}^{n} \frac{1}{\prod_{k=1}^{p} \mid x_{ik} - x_{jk} \mid^2} \tag{4.15}$$

where $\mathbf{D}$ are all the candidate designs, x is the design point with the first subscript as the index for the design points and the second subscript as the index for the dimension. It

was shown that computational time for a MaxPro design is comparable to other competing experimental designs. In Figure 4.5, an example of comparison between MaxPro design and the MmLHD in a three-dimensional space with 100 runs is provided. It is obvious that the MaxPro design has better coverage in the two-dimensional subspace compared with the MmLHD. Another benefit of using the MaxPro design is that it supports a sequential design in which we can add additional design points after the baseline design is established. With the capability of augmentation, we can preserve previous results from the baseline design points, and only a few new points are required to be investigated. In this study, we will adopt the MaxPro design as our choice for the design of experiments in the exploratory analysis.



(a) MaxPro Design (b) MaximinLHD Design

Figure 4.5: Comparison of MaxPro design and MaximinLHD in a 3D space with 100 runs

4.2.4 Surrogate models

Several surrogate models can be used in a computer experiment, and they are introduced in this subsection. We will design an experiment to test their performance under different scenarios to select the best surrogate model out of all surrogates mentioned here. The comparison will be presented in section 4.3.2.

Polynomial regression

Polynomial regression is a linear model with different combinations of polynomial terms as the predictors. A second-order polynomial is typically used in the application to capture the interaction effects and avoid overfitting. If we assume the response is y and there exist p factors from x_1 to x_p, then the prediction formula can be presented in following form:

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \sum_{j=1}^{p} \beta_{jj} x_j^2 + \sum_{k=1}^{p-1} \sum_{l=k+1}^{p} \beta_{lk} x_l x_k + \epsilon \tag{4.16}$$

Since it is a linear model, the regression coefficients β can be easily estimated using the least square method. When pairing with the design of experiments, a space-filling design may not perform well with this model due to Runge's phenomenon. Instead, a design of experiments in which the design points are pushed more toward the boundary or edges is preferred for this model.

Kernel regression with a Gaussian kernel

Kernel regression is a type of linear smoothers, and it is a non-parametric method. The prediction of a kernel regression is made through a weighted average of all the neighboring observations. The weights are determined by the distance from a new data point to all adjacent points. A higher distance would result in a lower value on the corresponding weight. A kernel function is typically used for modeling the weights, and a popular choice of the kernel function is in the form of Gaussian. The prediction formula is shown as follows:

$$\begin{aligned} \hat{y} = \hat{f}(\tilde{x}) &= \frac{\sum_{i=1}^{n} K(\tilde{x}, x_i) y_i}{\sum_{i=1}^{n} K(\tilde{x}, x_i)} \\ K(\tilde{x}, x_i) &= \exp\{-\theta \cdot \sum_{j=1}^{p} (\tilde{x}_{\cdot j} - x_{ij})^2\} \end{aligned} \tag{4.17}$$

where x's and y are the inputs and output from the experiment, $\tilde{x}$ is newly observed data, and θ is the bandwidth parameter from the Gaussian. The θ parameter can be estimated by minimizing the mean squared cross-validation error. Equation 4.17 is the result derived from the following minimization process:

$$\min_{\hat{y}} \sum_{i=1}^{n} K(x, x_i)\{y - \hat{y}\}^2 \tag{4.18}$$

For the case mentioned above, the $\hat{y}$ is a constant, and it is called the local constant estimator, which is the original form proposed by Nadaraya and Watson in 1964. By changing the $\hat{y}$ to a linear polynomial, the model would turn into a local linear estimator, and it becomes a more flexible surrogate compared to its counterpart.

Inverse distance weighting

Inverse distance weighting is also a non-parametric method, and it is similar to kernel regression. By replacing the Gaussian kernel with a reciprocal of a distance metric, the prediction can be expressed as follows:

$$\begin{aligned} \hat{y} = \hat{f}(\tilde{x}) &= \frac{\sum_{i=1}^{n} K(\tilde{x}, x_i) y_i}{\sum_{i=1}^{n} K(\tilde{x}, x_i)} \\ K(\tilde{x}, x_i) &= \frac{1}{d^2(\tilde{x}, x_i)} \end{aligned} \tag{4.19}$$

It can be shown that the inverse distance weighting model is an interpolator which means that there is no error on the points where the observations exist.

Radial basis function

Radial basis function (RBF) is a model based on a linear combination of basis functions, and unlike the polynomial regression, basis functions in RBF are not selected in advance, and they are dependent on the location of the data. Another property of these basis functions is that they are radial functions, which provide a mapping from $\mathbb{R}^p$ to $\mathbb{R}$. The formulation

of an RBF with a Gaussian basis function is shown as follows:

$$\begin{aligned} y &= \sum_{i=1}^{n} c_i B(x - x_i), \quad x \in \mathbb{R}^p \\ B(x - x_i) &= \exp\bigl(-\theta \cdot ||x - x_i||^2\bigr) \end{aligned} \tag{4.20}$$

To write it in the matrix form, assume that

$$\boldsymbol{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} B(x_1 - x_1) & \cdots & B(x_1 - x_n) \\ & \ddots & \\ B(x_n - x_1) & \cdots & B(x_n - x_n) \end{bmatrix}, \quad \boldsymbol{r}(x) = \begin{pmatrix} B(x - x_1) \\ \vdots \\ B(x - x_n) \end{pmatrix}$$

then Equation 4.20 can be expressed as

$$\boldsymbol{y} = \boldsymbol{R}\boldsymbol{c} \tag{4.21}$$

The coefficients $\boldsymbol{c}$ can be solved as

$$\hat{\boldsymbol{c}} = \boldsymbol{R}^{-1}\boldsymbol{y} \tag{4.22}$$

thus the prediction from the RBF is

$$\hat{f}(x) = \boldsymbol{r}(x)\hat{\boldsymbol{c}} = \boldsymbol{r}(x)\boldsymbol{R}^{-1}\boldsymbol{y} \tag{4.23}$$

To ensure a unique solution exists, $\boldsymbol{R}$ has to be a nonsingular matrix.

Gaussian process regression

Gaussian process regression, sometimes called Kriging, is a method developed by Matheron in 1963, and it was first used in the Geostatistics domain. The method follows a Bayesian framework in which a prior distribution is assumed for the function $f(x)$. With

data acquired from the experiment, we can make the prediction through the posterior distribution of the function $f(x)$. Here, the prior distribution is taken as a Gaussian Process (GP), which is essentially a multivariate normal distribution. The function $f(x)$ with the data observed from the experiment $(x_1, y_1), \cdots, (x_n, y_n)$ can be represented using the following expression:

$$\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix} \sim \mathcal{N}_n(\mu \mathbf{1}, \sigma^2 \boldsymbol{R}) \tag{4.24}$$

where μ and σ^2 are the mean and variance of the GP, $\boldsymbol{R}$ is the correlation matrix, and $\mathbf{1}$ is a column vector of 1. The joint distribution of $f(x)$ and the data observed can be expressed as

$$\begin{pmatrix} f(x) \\ y_1 \\ \vdots \\ y_n \end{pmatrix} \sim \mathcal{N}_{n+1} \left(\begin{pmatrix} \mu \\ \mu \mathbf{1} \end{pmatrix} \quad , \quad \begin{bmatrix} \sigma^2 & \sigma^2 \boldsymbol{r}(x)^T \\ \sigma^2 \boldsymbol{r}(x) & \sigma^2 \boldsymbol{R} \end{bmatrix} \right) \tag{4.25}$$

where the $\boldsymbol{r}(x)$ is the correlation vector between x and $x_1 \cdots x_n$. To derive the posterior distribution of $f(x)$, a useful lemma is stated as follows:

Lemma 4.2.1 *Suppose there exist random vectors V_1 and V_2 and they are of the size $m \times 1$ and $n \times 1$ respectively. If the joint vector of V_1 and V_2 follows a multivariate normal distribution,*

$$\begin{pmatrix} V_1 \\ V_2 \end{pmatrix} \sim \mathcal{N}_{m+n} \left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \quad , \quad \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

then the conditional distribution of V_2 on the V_1 is

$$V_1 | V_2 \sim \mathcal{N}_m(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(V_2 - \mu_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$$

Using this lemma, the posterior distribution of $f(x)$ can be derived as below:

$$f(x)|y_1, \cdots, y_n \sim \mathcal{N}\left(\mu + \boldsymbol{r}(x)^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \mu\mathbf{1}), \sigma^2\left\{1 - \boldsymbol{r}(x)^T \boldsymbol{R}^{-1}\boldsymbol{r}(x)\right\}\right) \tag{4.26}$$

With the data acquired from the experiment, the mean of the $f(x)$ changes from the assumed value μ. Also, the variance of the distribution, which is a measure of uncertainty, is reduced. The unknown parameters μ and σ^2 can be found using maximum likelihood estimates, and they have the following forms:

$$\begin{aligned} \hat{\mu} &= \frac{\mathbf{1}^T \boldsymbol{R}^{-1}\boldsymbol{y}}{\mathbf{1}^T \boldsymbol{R}^{-1}\mathbf{1}} \\ \hat{\sigma^2} &= \frac{1}{n}(\boldsymbol{y} - \hat{\mu}\mathbf{1})^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}) \end{aligned} \tag{4.27}$$

The correlation parameter θ can be estimated using the empirical Bayes method, and however, no explicit solution is found. The values have to be retrieved using an optimization process stated as follows:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \left[n \log \hat{\sigma^2} + \log |\boldsymbol{R}| \right] \tag{4.28}$$

To extend the Gaussian process regression from point data to functional responses in time, the original formulation can be modified by adding timestamps to the set of input variables. Assume the number of observations is n and the functional responses can be segmented into m evenly spaced portions; the prediction formula can be expressed as

$$\hat{f}(x,t) = \hat{\mu} + \tilde{\boldsymbol{r}}(x,t)^T \tilde{\boldsymbol{R}}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \quad \tilde{\boldsymbol{R}} \in \mathbb{R}^{mn \times mn} \tag{4.29}$$

The computation cost to inverse the $\tilde{\boldsymbol{R}}$ matrix is in the order of $O(m^3 n^3)$ and as the size dataset grows more significant along with a higher number of discretization points, this becomes a time-consuming task and even not tractable in practice. To avoid this issue, one approach is to assume the separability of the $\tilde{\boldsymbol{R}}$ matrix, and it can be separated into two

terms through the Kronecker product:

$$\tilde{\boldsymbol{R}} = \boldsymbol{R}_x \otimes \boldsymbol{R}_t \tag{4.30}$$

where $\boldsymbol{R}_x$ is the correlation matrix in the input variable space while $\boldsymbol{R}_t$ is the correlation matrix in the time domain. The inverse and determinant of $\tilde{\boldsymbol{R}}$ matrix can be also be separable and they are expressed as follows:

$$\tilde{\boldsymbol{R}}^{-1} = \boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_t^{-1} \tag{4.31}$$

$$|\tilde{\boldsymbol{R}}| = |\boldsymbol{R}_x|^m \otimes |\boldsymbol{R}_t|^n \tag{4.32}$$

Using the Kronecker product, the computation time reduces from the order of $O(m^3 n^3)$ to $O(m^3 + n^3)$. The prediction of the functional response can then be formulated as

$$\hat{f}(x,t) = \hat{\mu} + (\boldsymbol{r}_x(x) \otimes \boldsymbol{r}_t(t))^T \left\{\boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_t^{-1}\right\} (\boldsymbol{y} - \hat{\mu}\mathbf{1}) \tag{4.33}$$

where

$$\begin{aligned} \hat{\mu} &= \frac{\mathbf{1}' \left\{\boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_t^{-1}\right\} \mathbf{y}}{\mathbf{1}' \left\{\boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_t^{-1}\right\} \mathbf{1}} \\ \hat{\sigma^2} &= \frac{1}{mn} (\boldsymbol{y} - \hat{\mu}\mathbf{1})' \left\{\boldsymbol{R}_x^{-1} \otimes \boldsymbol{R}_t^{-1}\right\} (\boldsymbol{y} - \hat{\mu}\mathbf{1}) \end{aligned} \tag{4.34}$$

The correlation parameter can be estimated using the empirical Bayes approach as follow:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \left[mn \log \hat{\sigma^2} + m \log |\boldsymbol{R}_x| + n \log |\boldsymbol{R}_t| \right] \tag{4.35}$$

A model consisting of two Gaussian processes can be used for functional responses in unequal lengths. One Gaussian process is designed to capture the length information, while the other is dedicated to estimating the shape of a functional response. With the length and shape information at hand, we can recover signals in the time domain, and thus predictions

can be made for this type of response.

4.2.5 Sensitivity analysis

A surrogate model can be used for prediction and can be applied to investigate the impact of different input variables. In a sensitivity analysis, key variables which play a significant role in the response can be identified, and we can also estimate the interactions between various variables. In a linear regression model, we can determine important variables by comparing the magnitude of the coefficients. However, it cannot be applied when dealing with a surrogate model like Gaussian process regression. Sobol [85] proposed a methodology based on functional analysis of variance (ANOVA) to find the sensitivity estimates for a nonlinear model. Suppose a function $f(\boldsymbol{x})$ can be decomposed into the following form:

$$f(\boldsymbol{x}) = f_0 + \sum_{k=1}^{d} f_k(x_k) + \sum_{1 \le j < k \le d} f_{jk}(x_j, x_k) + \cdots + f_{1\cdots d}(x_1, \cdots, x_d) \tag{4.36}$$

With the assumption that zero expected value for $f_I(\boldsymbol{x}_I)$ with respect to $\boldsymbol{x}_k$ if index k is a subset of I,

$$\int f_I(\boldsymbol{x}_I) d\boldsymbol{x}_k = 0 \quad \text{for any } k \in I \tag{4.37}$$

the mean effect, main effect, and the two factor interaction can be shown as

- mean effect

$$\begin{aligned} \int f(\boldsymbol{x}) d\boldsymbol{x} &= \int_0^1 \cdots \int_0^1 f(\boldsymbol{x}) \prod_{k=1}^{d} dx_k = f_0 \\ f_0 &= \int f(\boldsymbol{x}) d\boldsymbol{x} \end{aligned} \tag{4.38}$$

- main effect (the subscript $-i$ means all x's except x_i)

$$\begin{aligned} \int f(\boldsymbol{x}) d\boldsymbol{x}_{-i} &= \int_0^1 \cdots \int_0^1 f(\boldsymbol{x}) \prod_{k \neq i} dx_k = f_0 + f_i(x_i) \\ f_k(x_k) &= \int f(\boldsymbol{x}) d\boldsymbol{x}_{-k} - f_0 \end{aligned} \tag{4.39}$$

- two factor interaction

$$\begin{aligned} \int f(\boldsymbol{x}) d\boldsymbol{x}_{-i,-j} &= \int_0^1 \cdots \int_0^1 f(\boldsymbol{x}) \prod_{k \neq i,j} dx_k = f_0 + f_i(x_i) + f_j(x_j) + f_{ij}(x_i, x_j) \\ f_{ij}(x_i, x_j) &= \int f(\boldsymbol{x}) d\boldsymbol{x}_{-i,-j} - f_0 - f_i(x_i) - f_j(x_j) \end{aligned} \tag{4.40}$$

In the context of the functional ANOVA, total variance for the function $f(\boldsymbol{x})$ can be expressed as follow:

$$\begin{aligned} V_{total} &= \mathbb{E}[(f(\boldsymbol{x}) - f_0)^2] \\ &= E(f^2(\boldsymbol{x}) - 2f(\boldsymbol{x})f_0 + f_0^2) = \mathbb{E}[f^2(\boldsymbol{x})] - f_0^2 \end{aligned} \tag{4.41}$$

The variance for a subset of variables $I \subset \{1, ..., d\}$ can be represented as

$$V_I = \mathbb{E}[(f_I(\boldsymbol{x}_I) - \mathbb{E}[f_I(\boldsymbol{x}_I)])^2] = \mathbb{E}[f_I^2(\boldsymbol{x}_I)] \tag{4.42}$$

From Equation 4.41 4.42, the total variance can be decomposed as

$$
\begin{aligned}
V_{total} &= \int f^2(\boldsymbol{x})d\boldsymbol{x} - f_0^2 \\
&= \int \left[f_0^2 + \sum_{i=1}^{d} f_i^2(x_i) + \sum_{i\neq j} f_{ij}^2(x_i, x_j) + \cdots + f_{1,\ldots,d}^2(x_1, ..., x_d) + \cdots \right] d\boldsymbol{x} - f_0^2 \\
&= \int \sum_{i=1}^{d} f_i^2(x_i)d\boldsymbol{x} + \int \sum_{i\neq j} f_{ij}^2(x_i, x_j)d\boldsymbol{x} + \cdots + \int f_{1,\ldots,d}^2(x_1, ..., x_d)d\boldsymbol{x} \\
&= \int \sum_{i=1}^{d} f_i^2(x_i)d\boldsymbol{x} + \sum_{i\neq j} \int f_{ij}^2(x_i, x_j)d\boldsymbol{x} + \cdots + \int f_{1,\ldots,d}^2(x_1, ..., x_d)d\boldsymbol{x} \\
&= \sum_{i=1}^{d} V_i + \sum_{i\neq j} V_{ij} + \cdots + V_{1\ldots d}
\end{aligned}
\tag{4.43}
$$

The Sobol's sensitivity indices are defined as

$$
S_I = \frac{V_I}{V_{total}} \quad \text{with} \quad \sum_I S_I = 1 \tag{4.44}
$$

By comparing the magnitude of these indices, the variables that significantly contribute to the response can be identified.

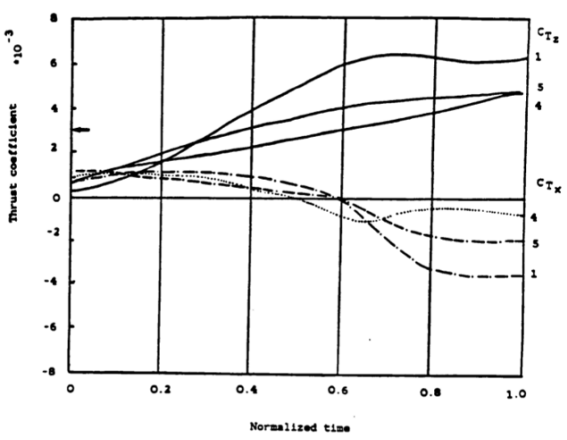
4.3 Experiments for the exploratory analysis

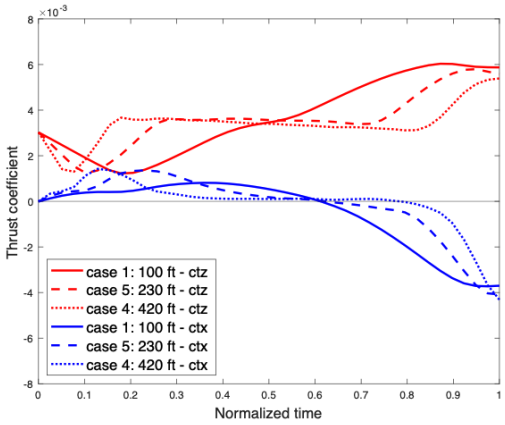
In this section, the simulation, which consists of a mathematical model with an optimization process, will be verified to ensure that it can predict the optimal control in autorotation maneuvers. To find an appropriate surrogate model for the exploratory analysis, various known functions in either low or high-dimensional space are used to test the strength of the surrogates mentioned in section 4.2.4. Once the surrogate model is selected, two metrics proposed to determine the adequate run size of the experiment are tested. We will use the metric to pick the proper run size for building surrogates to construct safety envelopes under various combinations of initial conditions.

For testing the effectiveness of the surrogate proposed for predicting functional responses in unequal lengths, an experiment that compares the prediction from the surrogate and the results from the simulation will be described. Finally, we will conduct a sensitivity analysis to identify the key parameters that play a significant role in affecting the shape of the envelope.

4.3.1 Model verification

As mentioned in Section 4.1, the model is required to be verified either through physical experimentation or through the results from other comparable simulations. The simulation of computing the optimal control for the autorotation consists of a two-dimensional helicopter model in a vertical plane with an optimization process based on direct collocation. In Figure 4.6, the trajectories of computed controls for different entry altitudes are compared with the ones found in [42]. Here the entry speed is kept at 12 knots, but the entry heights can vary among 100 feet, 230 feet, and 420 feet. These curves follow a similar trend and vary within the same domain, but the calculated flight times are different, especially for the case starting at 420 feet. The flight times computed in this study are generally larger, and they are monotonically increasing with higher altitude. In Figure 4.7, the calculated controls for different entry speeds are compared with the same source. The entry height remains constant at 100 feet, but the entry speed can change from 12 knots, 38 knots, to 57 knots. Again, the profiles of controls look similar to each other, but the flight times computed are slightly shorter in this scenario. The probable causes for the differences between the results from the literature and the ones from current implementation may be attributed to (1) a different objective function and (2) the addition of rate controls in the formulation for mitigating the possibility of having highly fluctuated controls.





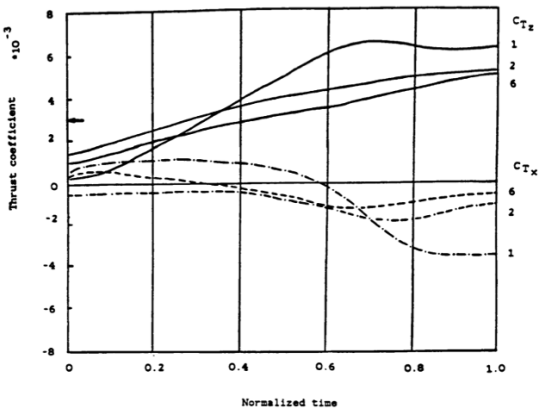
ENTRY CONDITION	1	5	4
ENTRY HEIGHT (ft.)	100	230	420
FLIGHT TIME (sec.)	5.5	9.3	11.3

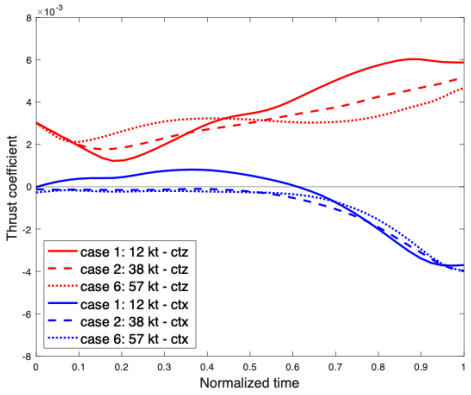
ENTRY CONDITION	1	5	4
ENTRY HEIGHT (ft.)	100	230	420
FLIGHT TIME (sec.)	5.87	10.16	16.54

Results from literature [42]

Results from current implementation

Figure 4.6: Comparison of results for different altitudes from the literature





ENTRY CONDITION	1	2	6
ENTRY SPEED (Knots)	12	38	57
FLIGHT TIME (sec.)	5.5	8.7	10.4

ENTRY CONDITION	1	2	6
ENTRY SPEED (Knots)	12	38	57
FLIGHT TIME (Sec.)	5.87	6.5	9.32

Results from literature [42]

Results from current implementation

Figure 4.7: Comparison of results for different forward speeds from the literature

The typical outputs of the simulation are shown in Figure 4.8, and they include several signals, such as the state and control variables. This is a test case with an initial altitude of 100 feet and an initial speed of 12 knots. The state variables which describe the vehicle behavior are displayed on the top row of the figure, and the control variables are presented at the bottom. From the angle of the tip-path-plane, which is one of the controls exerted to the dynamical system, we can see that the control strategy employed here is first to speed up the vehicle and then drastically decrease the speed before touchdown. To build a surrogate model, we need to acquire the following signals: the terminal horizontal and vertical speeds, and control trajectories of C_T and α. They are marked using a red box or circles in Figure 4.8. The terminal speed is primarily used for envelope exploration, while the controls obtained from the simulation are subjected to building the surrogate for predicting the optimal control for unobserved conditions.

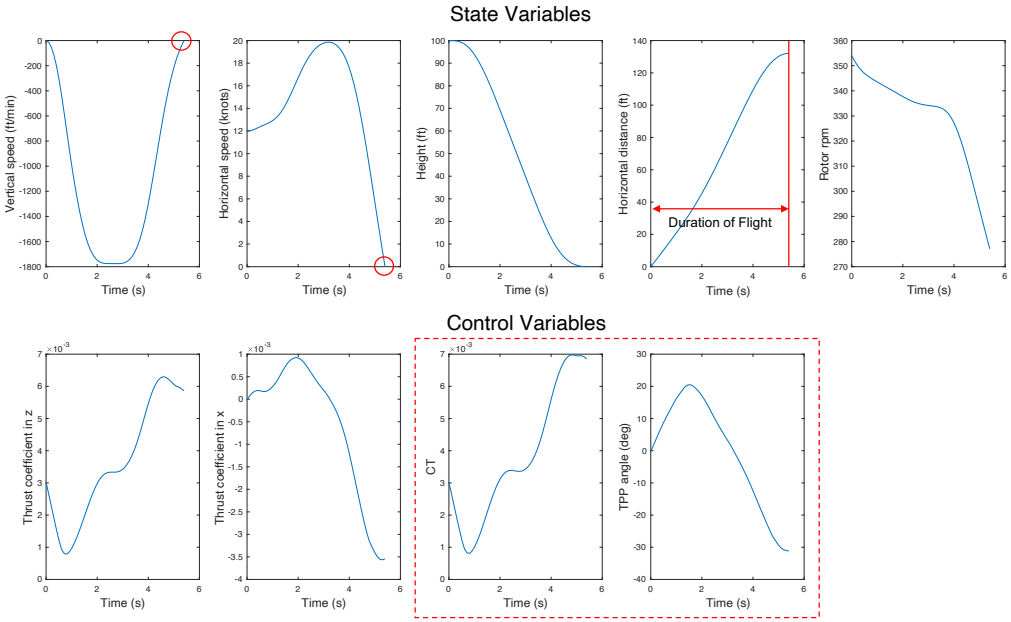


Figure 4.8: Typical outputs from the computer code

4.3.2 Surrogate model selection

In section 4.2.4, we have introduced several surrogate models that can be used for computer experiments. An experiment (E4-A) is required to find the appropriate one to support the exploration of the operational space. In [86], Surjanovic et al. collected a variety of test functions, and they can be used for many purposes such as optimization, prediction, and uncertainty quantification, etc. In this test, we have selected three functions in two-dimensional space (from [86] [87] [88]) and also chosen another four functions in higher dimensional space (from [86]). The visualization of functions selected in two-dimensional space is shown in Figure 4.9. We chose to use the MaxPro design with 81 runs to place the experimental points in the space to be explored. However, for the surrogate of polynomial regression, a space-filling design is not ideal because it would lead to a higher oscillation in predictions at the boundary. Thus, the custom design that pushes design points to the boundary was selected for this particular surrogate rather than using the MaxPro design. The locations of the design points in this custom design are picked based on Chebyshev nodes, which can be obtained using the following formula:

$$d_k = \cos\left(\frac{2k-1}{2n}\pi\right), \quad k = 1, \cdots, n \quad \text{and} \quad d_k \in [-1, 1] \tag{4.45}$$

where n is the number of points desired to be placed in each dimension. The designs of experiments used in the experiment are displayed in Figure 4.10.

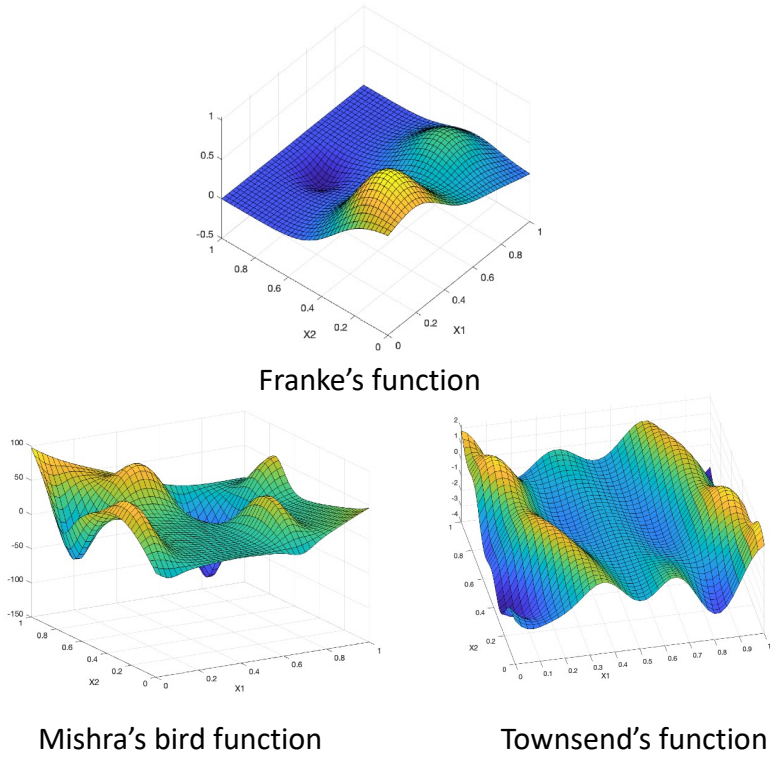


Figure 4.9: Two-dimensional functions used for testing the surrogates

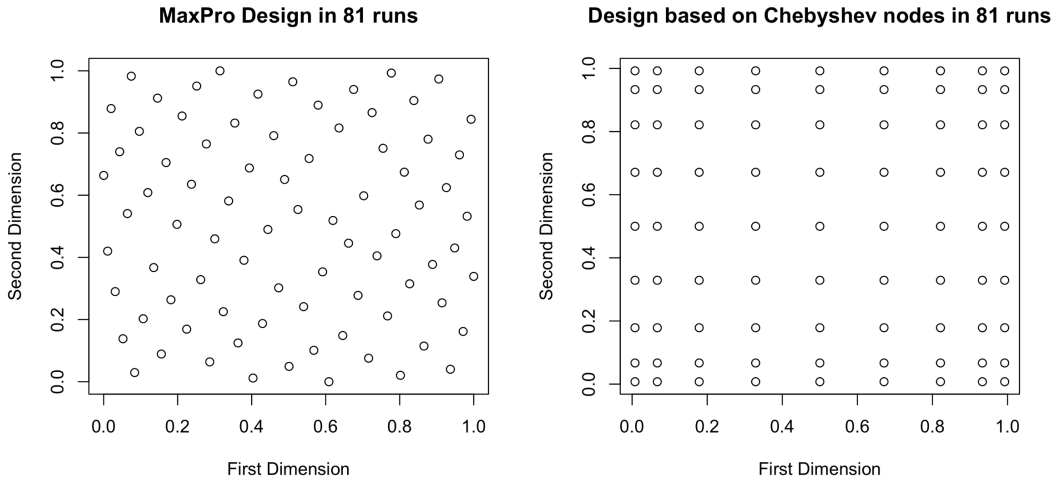


Figure 4.10: Designs of experiments used for testing the surrogates

To measure the performance of a surrogate, a thousand points that are different from the design points are generated using Sobol sequences, and the predictions on these thousand

points from the surrogate are compared with the actual values from the true function. Here we chose the root mean square error (RMSE) as the metric for accuracy. The results of the experiment on two-dimensional functions are shown in Figure 4.11. Gaussian process regression consistently performs the best in the group, followed by radial basis function. Polynomial regression is ranked in third place, but owing to the need for a custom design and not easy to determine the terms to include for higher dimensional situations, we eliminate this option for the following test on the high dimensional functions. Figure 4.12 shows the results of comparing the performance of Gaussian process regression, radial basis function, and kernel regression on high dimensional functions. Gaussian process regression outperformed the rest in the group for all functions tested except the six-dimensional function (OTL circuit function). In sum, Gaussian process regression was selected as the surrogate model used in the exploratory analysis.

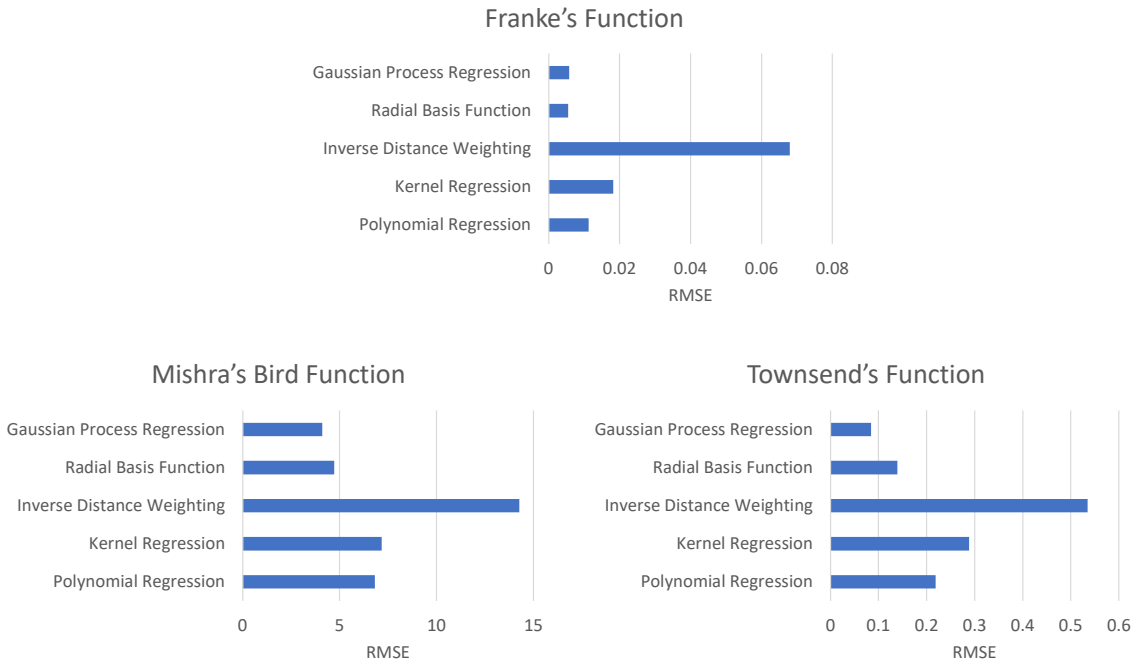


Figure 4.11: Results of comparing different surrogates among two-dimensional functions

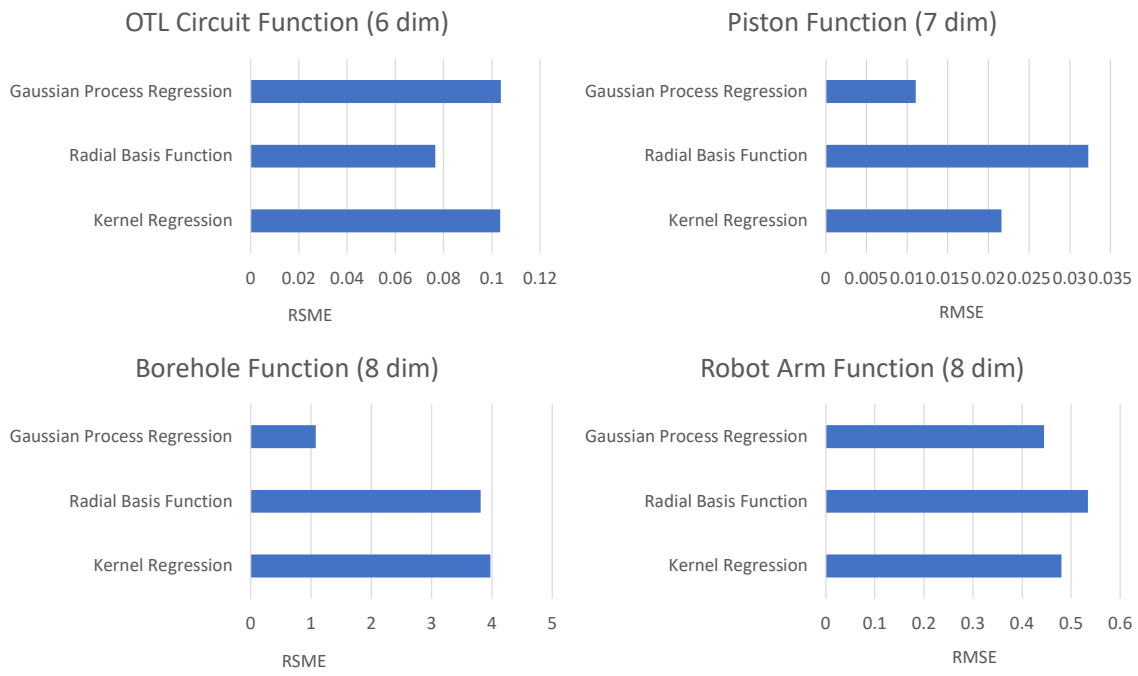


Figure 4.12: Results of comparing different surrogates among high dimensional functions

4.3.3 Determining the run size for an experiment

Given the type of surrogate and the kind of design of experiments selected, we must also determine the number of runs in the design region before the experiment. A surrogate built upon fewer runs may be insufficient to capture the details of the actual function and may have limited prediction capability. In the meantime, it is not prudent to blindly increase the number of runs because it may drastically increase the time for computation, especially for resource-intensive simulations. An appropriate number of runs for the experiment should strike a balance between accuracy and efficiency. It is a challenging task because in the real scenario, the behavior of an actual function is unknown, and we can only accumulate the knowledge or information of this actual function through performing experiments. To tackle this task, Loeppky et al. [89] conducted a study on selecting the sample size for an experiment, and they suggested that the initial run size should be at least ten times the dimension of the input variables. In this experiment (E4-B), we will use this value as the initial run size and sequentially add more runs to this baseline design. With the

availability of an actual function, we can measure the benefit of conducting additional runs by calculating the difference between the prediction and the actual function. Nevertheless, this performance index cannot be obtained in real life because we do not know the behavior of a system in advance. Therefore, we propose two metrics that could be helpful to infer the trend of the performance index, i.e., the prediction error:

(1) M_1: the sum of absolute differences between the predictions from a surrogate built upon the baseline design and the predictions from another surrogate constructed using a design with additional runs

(2) M_2: the sum of absolute differences between predictions from two successive designs in terms of run size

The idea behind the first metric M_1 is that we expect the difference between the predictions from a surrogate built upon the baseline design and the predictions from another surrogate constructed using a design with additional runs would initially increase with the run size and then eventually level off. This M_1 metric should follow a similar profile of the actual prediction error, but instead of decreasing with more runs, it would show a growing trend. The second metric M_2 is used to monitor the improvement with each additional run. We expect the difference in predictions between successive runs would decrease when the solution convergence is reached.

To test the proposed metrics, the experiment is set up as follows:

1. Initialize the run size as ten times the dimension of the input variables and choose a value for the maximum number of additional runs. For illustration purposes, an example of a system that has two input variables is shown in Figure 4.13. The baseline design would have twenty runs in this case, and the locations for these runs are displayed as black circles. Also, the maximum number of additional runs is selected as a hundred, and the locations for conducting these runs are depicted in red numbers.

2. For evaluation, another five hundred points different from the design points mentioned above are produced using Maximin Latin Hypercube Design (MaximinLHD), and their locations are shown as green crosses in Figure 4.13. With these evaluation points, we can compute the actual prediction error, the proposed M_1, and M_2 metrics.

3. Compare the trends between the proposed metrics and the actual prediction error under some known functions in low and high dimensional space. If the profiles of these metrics have a similar pattern as the ones for the actual prediction error, we can use them to guide the selection of the run size for an experiment.

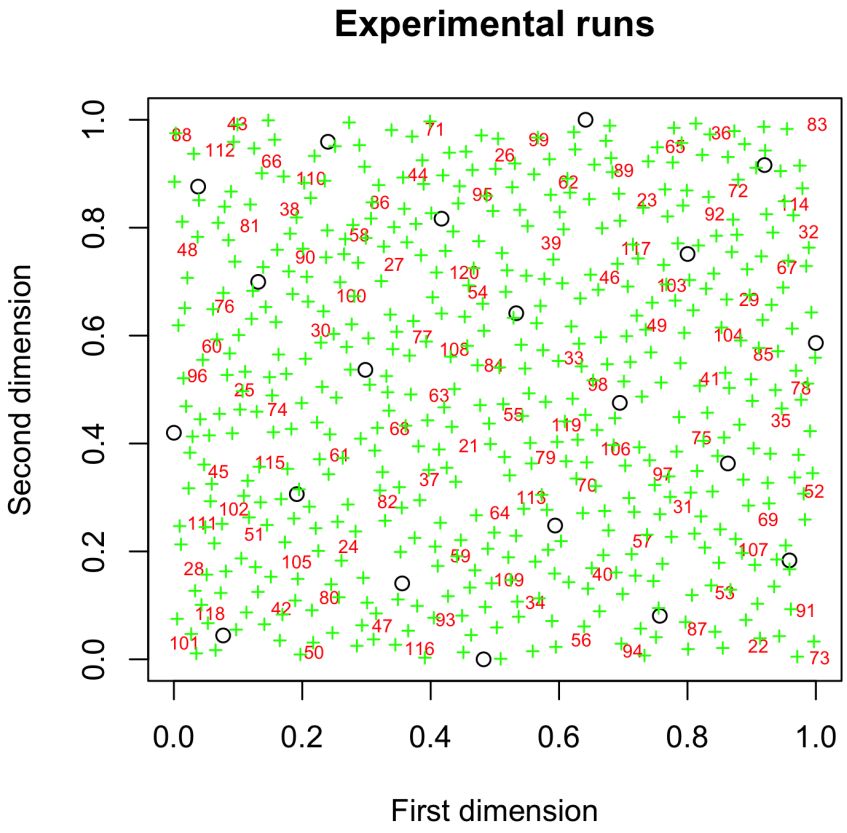


Figure 4.13: The setting of the experiment for testing the metrics on run size selection

The results of the tests for low and high dimensional functions on different run sizes are shown separately in Figure 4.14 and 4.15. The actual prediction error is shown on the top row in each figure, while the M_1 and M_2 metrics are displayed in the middle and the bottom rows, respectively. From the observation, the M_1 metric levels off approximately around the same number of runs as the actual prediction error flattens out, so it can serve as a proxy to infer the trend. For the M_2 metric, we hypothesize that it would have a decreasing trend, but this phenomenon only was observed for some low dimensional functions. Most of the M_2 curves obtained in the tests fluctuate as the number of runs grows, and it is difficult to use this metric to pinpoint the ideal run size. Therefore, the M_1 metric is selected to provide guidance on the determination of a proper run size for an experiment.

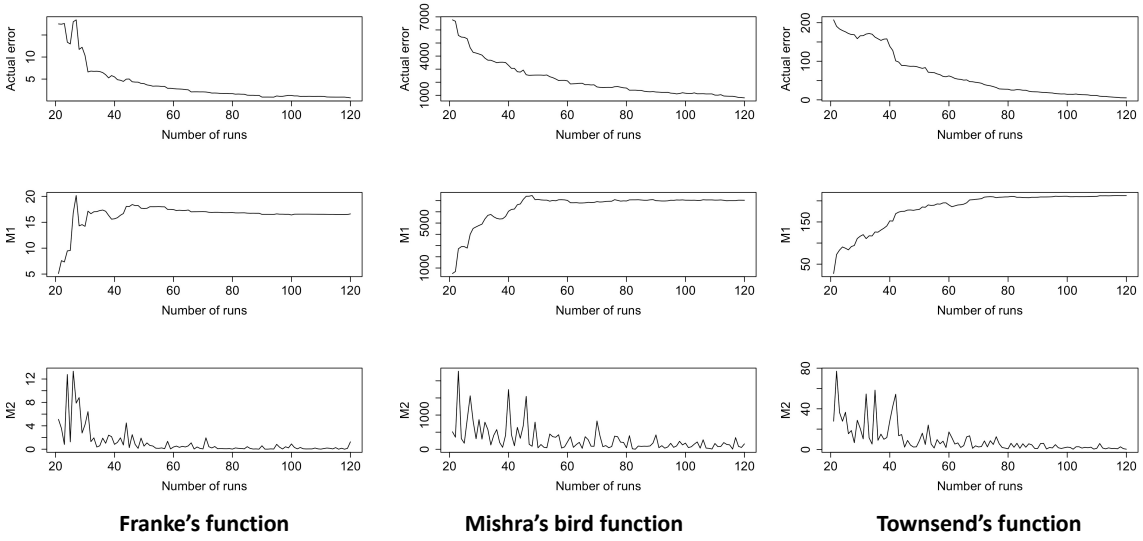


Figure 4.14: Test results for low-dimensional functions

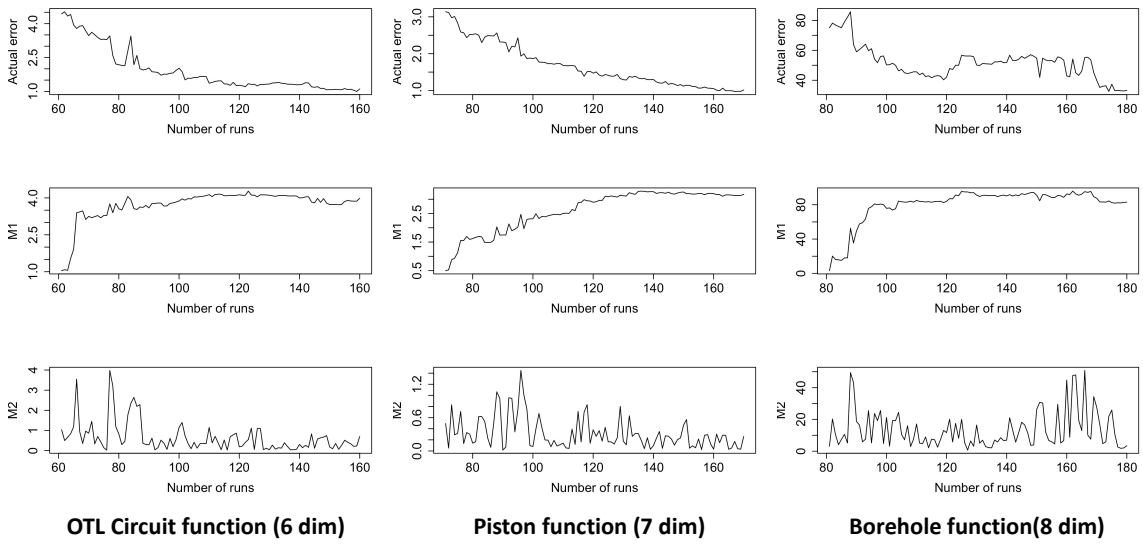
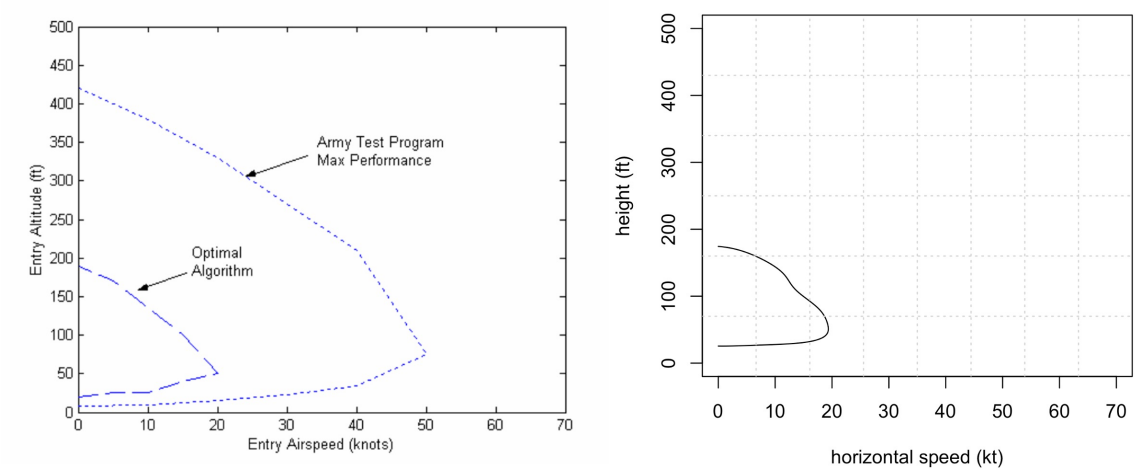


Figure 4.15: Test results for high-dimensional functions

4.3.4 Predicting safety envelopes under various conditions

With the experiments (E4-A and E4-B) conducted in previous sections, an appropriate surrogate model with the corresponding design of experiments and the metric for selecting proper run size are determined. For exploring the safety envelope of autorotation maneuver, a Gaussian process regression with 100 runs of MaxPro design is used in the investigation. The response to be predicted from the surrogate is the terminal speed. Based on [42], the acceptable touchdown forward speed for OH-58A is 3 knots, and the allowable terminal vertical speed is 8 feet per second. Thus, the boundary of the envelope will be drawn based on the value of 10 feet per second on the terminal speed. To verify the envelope found from our implementation, we compared the results with the safety envelope predicted from [47]. The side-by-side comparison is shown in Figure 4.16. From the left subplot, we can see that the envelope predicted using optimal control is drastically smaller than the one found in the flight test. Moreover, the envelope computed in our current implementation (shown in the right subplot) is similar to the one found in the literature. However, one difference observed between these two envelopes is that they have different smoothness. The one

found in the literature seems to be constructed using multiple consecutive line segments formed by representative points, while the one computed from the surrogate has a more smooth and continuous shape.



Safety envelope predicted from literature [47] for OH-58A

Safety envelope predicted from surrogate modeling

Figure 4.16: Verification of the safety envelope obtained from the surrogate model

To demonstrate the capability of using surrogate modeling for predicting safety envelopes in various scenarios, a few parameters were selected and varied to investigate their effects on the shape of the envelope. In the test, the region of interest in the height-velocity space is spanned by $H \in [0, 400]$ feet and $V \in [0, 60]$ knots. Three cases with different varying parameters are examined:

1. Initial descent velocity is varied from 0 to 300 feet per minute
2. Blade moment of inertia is varied from 236 to 436 slug-ft^2
3. Vehicle weight is varied from 3000 to 3400 lbs

The results of the experiment from these three cases are shown in Figure 4.17, 4.18, and 4.19, respectively. For the case with different entry descent speeds, the envelopes have a similar shape from hover to a descent rate up to 200 feet per minute. If the descent rate continues to increase from 200 to 300 feet per minute, a new restriction zone would emerge

around the corner with high entry forward speed and low entry altitude. For the case of different blade moments of inertia, a higher value of inertia would result in a smaller area in the restriction zone. We can interpret it as with a higher blade moment of inertia, the rotational energy is better kept during the process; thus, it is easier to speed up or slow down the vehicle as necessary. It is expected that with an even higher blade moment of inertia, the restriction zone would disappear, and the entire region would be free of risk. For the case with different vehicle weights, it is observed that the restricted area would be enlarged in the instance of having a heavier vehicle.

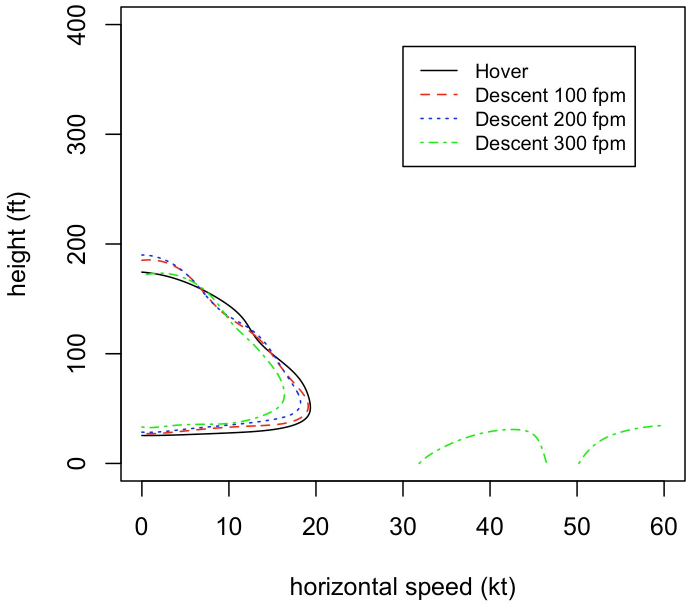


Figure 4.17: Safety envelopes for different entry descent speeds

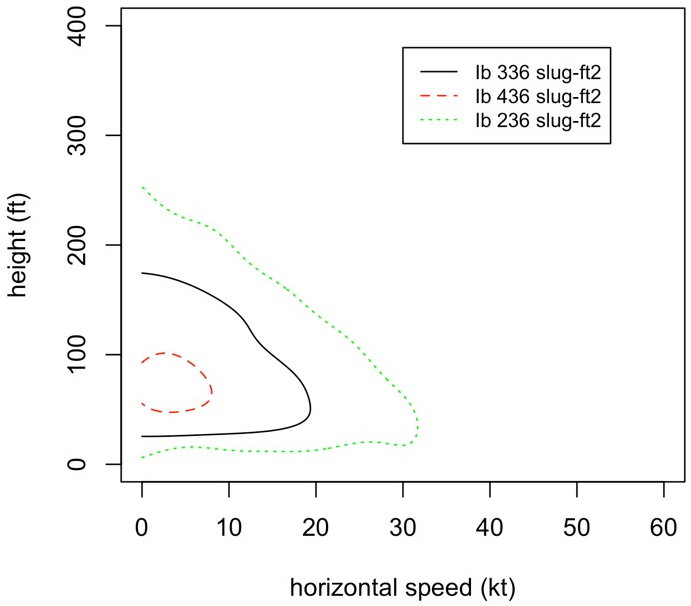


Figure 4.18: Safety envelopes for different blade moments of inertia

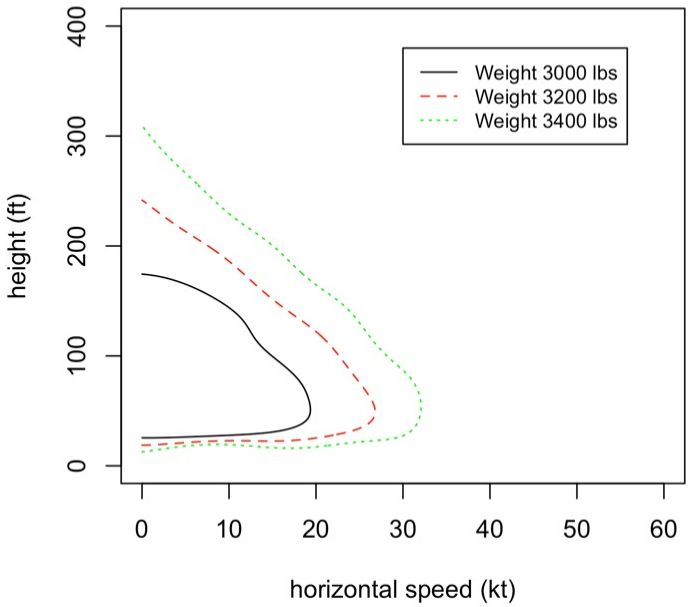


Figure 4.19: Safety envelopes for different vehicle weights

4.3.5 Predicting the required controls for recovery in an autorotation

In our exploratory analysis for a hazardous event, we intend to use a surrogate not only for finding the safety envelope but also for predicting the action for recovery when encountering the hazardous event. In section 4.2.4, a surrogate model for predicting functional responses in unequal lengths is proposed. Here, the method will be tested on predicting the optimal control for the autorotation maneuver. The setup of the experiment (E4-C) is stated as follows:

(1) A baseline design is constructed in the operational space to be explored, and the simulations on these design points are conducted to acquire the corresponding optimal controls. These control signals are in the form of functional responses.

(2) Use the data acquired from the previous step to build the surrogate model. This model consists of two Gaussian processes that capture both length and shape information of functional responses.

(3) Find a location in the design space where it is far away from all observed points and then use the surrogate to predict the functional responses on this specific location. For verification purposes, we will compare the prediction from the surrogate with the results obtained from the simulation.

In the experiment (E4-C), the baseline design is chosen as a 20-runs MaxPro design, and the distribution of the design points is shown as black circles in Figure 4.20. The red cross in the figure is the run 21 and its location ($h_0 = 159$ feet and $u_0 = 27$ knots) is selected to be far away from all design points in the space. The ranges of the entry horizontal speed and the entry height are set as $[0, 400]$ feet and $[0, 60]$ knots, respectively. The computed optimal control trajectories on these design points from the simulation are thrust coefficients in x and z directions, and they are displayed in Figure 4.21. The curves colored in red on the left are the control presented in the time domain, while the curves colored in blue on the

right are the ones shown with respect to normalized time. The normalized version of these control trajectories exhibit different shape patterns based on different initial conditions, and one Gaussian process regression is dedicated to capturing this information. The results of the experiment is shown in Figure 4.22, which is essentially a comparison between the predictions from the surrogate and the results from the simulation. We can see that even with a relatively small run size of 20, the proposed surrogate model can accurately predict the functional responses in unequal lengths.

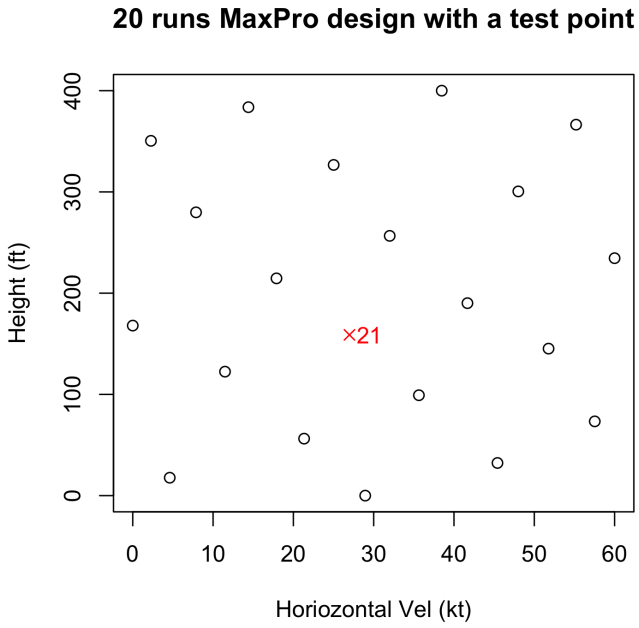


Figure 4.20: The setup of experimental runs for testing the predicted controls

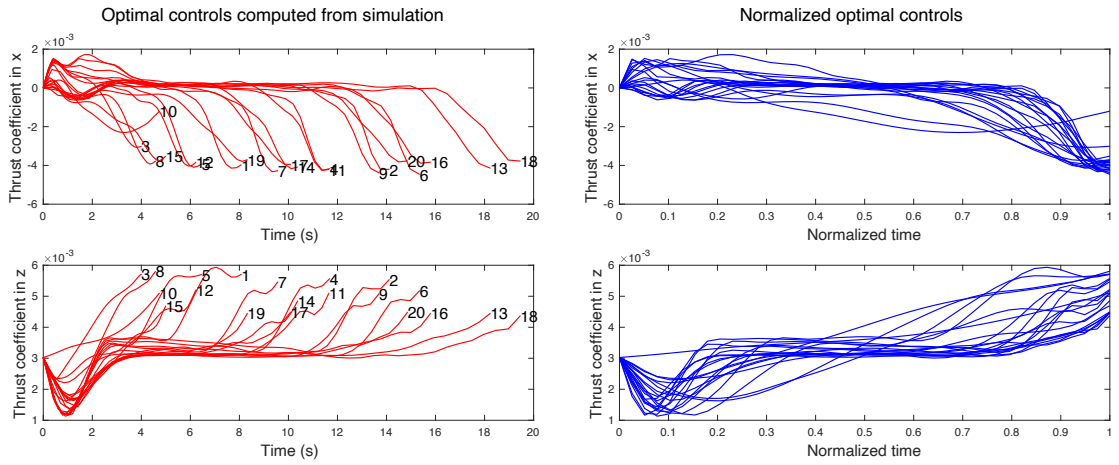


Figure 4.21: Data acquired from experimental runs for building the surrogate

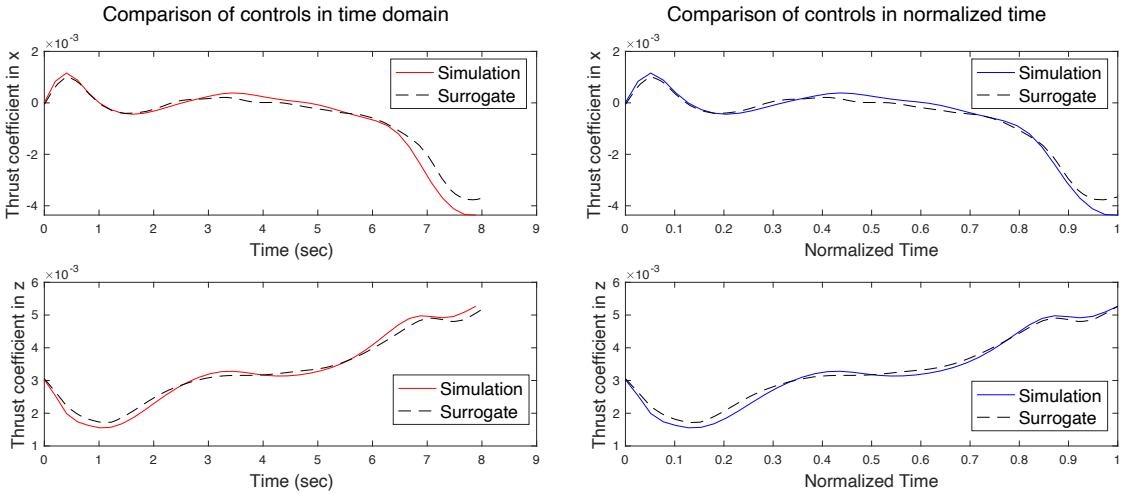


Figure 4.22: Comparison of results from the surrogate and from the simulation

4.3.6 Identifying the key variables in the operational space

To investigate the relative importance of the input variables, a sensitivity analysis built upon a surrogate model can be employed for identifying the key parameters. In the experiment (E4-D), three variables are included in the set of input variables: the entry height, the entry speed, and the vehicle's weight. The results from the experiment are shown in Figure 4.23 and 4.24. In the main effect plot, we can see that the horizontal velocity and the entry speed have a higher impact on the terminal speed than the vehicle's weight. Also, the height and

the horizontal speed have a nonlinear effect on the response, while the weight only has a linear relationship with the response. In terms of Sobol sensitivity indices shown in Figure 4.24, the entry speed has the highest value among all other variables, which means that it is the deciding factor for the response. The interactions between all combinations of input variables are shown in a heatmap. It can be seen that the interaction between the entry height and the entry speed is important, and the impact of this interaction is higher than the main effect of entry height.

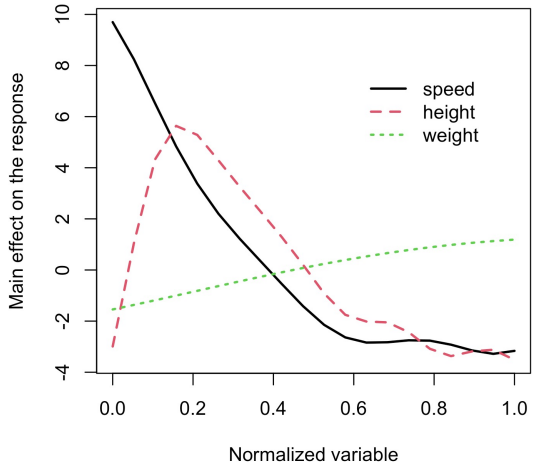


Figure 4.23: Main effect plot for the three input variables considered in the experiment

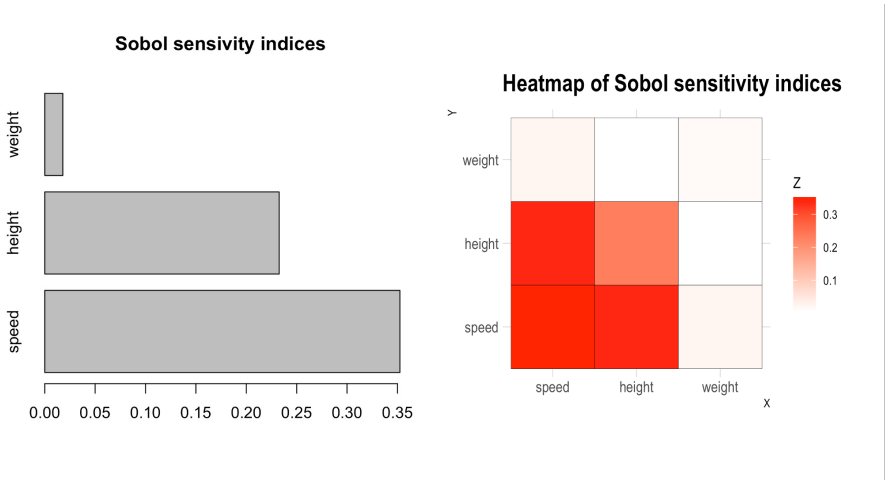


Figure 4.24: Sobol's sensitivity indices and the interaction effects on the response

4.4 Summary for the fourth research question

The exploratory analysis is a more proactive approach in addition to the retrospective analysis for enhancing the safety of rotorcraft operations. It aims to find the safety envelope of a hazardous event and predict the optimal controls in the operational space. A framework for efficiently tackling these two tasks was proposed, and we extended the idea of surrogate modeling from exploring the design space to investigating the operational space. We addressed four research questions regarding surrogate modeling with the design of experiments on operational space exploration. In the test, it was found that Gaussian process regression with a MaxPro design is an ideal surrogate for capturing the behavior of low and high dimensional test functions; thus, it is suitable for modeling the response in the operational space. To decide the proper run size for an experiment, the M_1 metric, which can serve as a proxy of the true prediction error, is viable to provide guidance on the selection. To predict the functional responses in unequal lengths, such as the optimal control trajectories, a surrogate with two Gaussian processes was proposed. The results show that the predictions from the surrogate match the ones from the simulation. In the end, a sensitivity analysis, which is built upon the surrogate created, can identify the key parameters for affecting the response. In the case of considering three input variables for the autorotation maneuver, we found that the entry height and the interaction between entry height and entry speed are the dominant factors to affect the terminal speed.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

From recent safety reports, helicopter accident statistics have an increasing trend, and more attention needs to be directed to the rotorcraft domain. As a strategy for improving the safety of rotorcraft operations, it is vital to conduct post-flight analysis on flight data records and then provide retrospective feedback to pilots or the management team on potential anomalous events for risk mitigation purposes. From the literature, most of the analyses on anomaly detection found in the studies were performed on flights in fixed-wing commercial aviation rather than helicopter flights in general aviation. To analyze flight data records specific to rotorcraft operations, we first identified the need for flight phases detection, and it is a prerequisite for the anomaly detection pipeline. To start with identifying flight phases, we conducted a survey on literature regarding the definitions of flight phases. Next, a set of baseline definitions of flight phases for rotorcraft was established through transcribing the existing definitions and combining opinions from subject matter experts. Several evaluation criteria were suggested to judge the identification results, and we can use them to determine how well the methods perform qualitatively. In the implementation, a logic that separates flight data records into two regions was proposed, and the associated flight phases for each region were specified. We introduced several candidate methods for identifying the flight phases in the high-altitude region. To find the best or a combination of methods, an experiment for comparing these methods was conducted. In the test, we found that the piecewise linear regression (PLR) with sequence smoother and the sliding window regression classification (SWRC) have the highest accuracy in prediction on a labeled dataset. The flight phases in the low-altitude region are relatively well-defined in

the feature space. Thus, the identification is performed using a filtering approach based on modified thresholds on flight parameters. Combining the labels detected from these two regions, the flight phase information of the entire flight can be retrieved, and the results satisfy the evaluation criteria mentioned above. As an extension to the basic flight phase identification algorithm, turn maneuvers, different takeoffs, or other maneuvers can be added as an augmentation to the baseline. For the takeoff identification, simulated trials were used to support the prototyping of the algorithm. Through the verification with another set of simulated runs, the algorithm for detecting takeoffs is validated. The overall process for identifying phases of flight for rotorcraft operation is summarized in Figure 5.1.

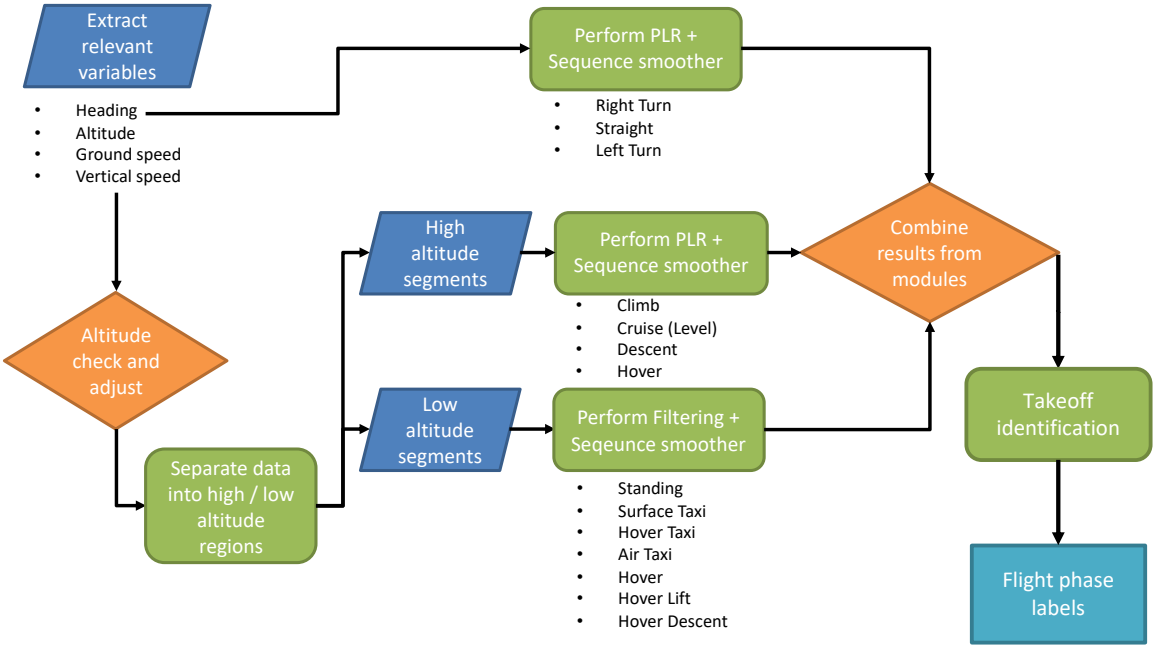


Figure 5.1: Summary of the process for flight phases identification in rotorcraft operations

With the accessibility of flight phase labels in an unlabeled dataset, we continued to develop a methodology for detecting anomalies on flight segments within the same flight phase of a routined helicopter operation. Here we characterize the anomalies as rare events and have different patterns. In the retrospective analysis, a sequential approach was proposed to detect different tiers of outliers. It consists of several elements, including trajectory pattern mining, time series length analysis, and shape analysis. In trajectory pattern mining,

we introduced several distance metrics with hierarchical clustering for pattern grouping. In the time-series shape analysis, several feature extraction and outlier detection methods were investigated for detecting anomalies on flight segments in comparable lengths. A set of experiments is required to compare the methods and guide the down-selection process.

To test the methods in the portfolio, we created synthetic and simulated data with known labels under various scenarios. A subset of methods in the portfolio appropriate for anomaly detection was identified in the experiment. Ultimately, we applied these methods to the actual initial climb and the approach segments in an air ambulance operation. Some highlights from this research topic are summarized as follows:

- To account for the conundrum of verifying results from unsupervised learning methods, synthetic and simulated data were created to evaluate the performance of methods and select the candidates out of a portfolio.

- For trajectory pattern mining, using Euclidean distance as the distance measure along with hierarchical clustering for pattern grouping is a viable approach for placing trajectories into relevant groups.

- To address the issue of comparing segments with unequal lengths, the segments were filtered into comparable lengths prior to the time series shape analysis.

- To detect shape anomalies for flight segments within comparable lengths, functional principal component analysis (FPCA) and convolutional variational autoencoder (CVAE) are useful for capturing the shape information using low-dimensional features.

- Compared to traditional exceedance analyses, the proposed methodology can detect potential anomalies without specifying thresholds on flight parameters.

In addition to the retrospective analysis mentioned above, we developed an exploratory analysis for investigating the safety envelope and the necessary recovery action for a hazardous event. Traditional methods for finding the optimal control rely on optimizing a

designed objective function. Depending on the combinations of model complexity and the optimization process, finding a solution may be time-consuming. In the framework, surrogate modeling is introduced to facilitate the construction of safety envelopes and the computation of recovery actions.

To demonstrate the feasibility of the framework, the autorotation, which is ranked at the top two occurrence categories in accidents, is chosen as a use case in this study. We selected a widely used mathematical model for simulating the autorotation. By combining this model with an optimal control solver, the control trajectories which encourage a safe landing can be obtained. For verification, we compare the outcomes from our implementation to the results found in the literature. In general, these solutions are consistent with each other. Before exploring the operational space of the autorotation, we conducted an experiment to find the appropriate combination of a surrogate model and a design of experiments. It was found that the Gaussian process regression with the MaxPro design is an adequate method for tackling functions in low and high-dimensional space. Further, a metric that could potentially support the decision-making on the number of runs required for an experiment was suggested. With the knowledge of picking the surrogate model and determining the proper run size, we investigated the safety envelopes for the autorotation under various scenarios, such as different entry descent speeds, blade moments of inertia, and vehicle weights.

To predict the optimal control trajectories in the operational space, we developed a surrogate model with two Gaussian processes, one to capture the length information and the other to learn the shape information. From the results of an experiment for validating this approach, the predicted controls from the surrogate in an unobserved location are similar to those derived from the simulation. Finally, we conducted a sensitivity analysis on the surrogate to investigate key input parameters in a selected set. It was observed that the entry speed and the interaction between the entry speed and the entry height play significant roles in affecting the magnitude of the terminal speed, which directly links to the outcome of an

autorotation maneuver.

5.2 Recommendations and future work

Concerning the work on detecting flight phases for rotorcraft operations, we can display different visualizations and statistics of flight phase information on a dashboard. This representation could assist pilots and the management team in looking at the flights from a new perspective. In Figure 5.2, an example of a potential dashboard implementation with statistics on flight phases is presented. Based on the information provided, the relevant parties could realize the time duration spent in each of the flight phases, and the major transitions between flight phases are. Further, other homogenous flight segments which are not discussed in this study, such as different types of approach maneuvers and traffic patterns, can be added to the current implementation for completeness. Moreover, with the arrival of eVTOL in the air taxi industry, it would be interesting to see how our current approach can be applied or adapted to fit the needs for a different aerial architecture.

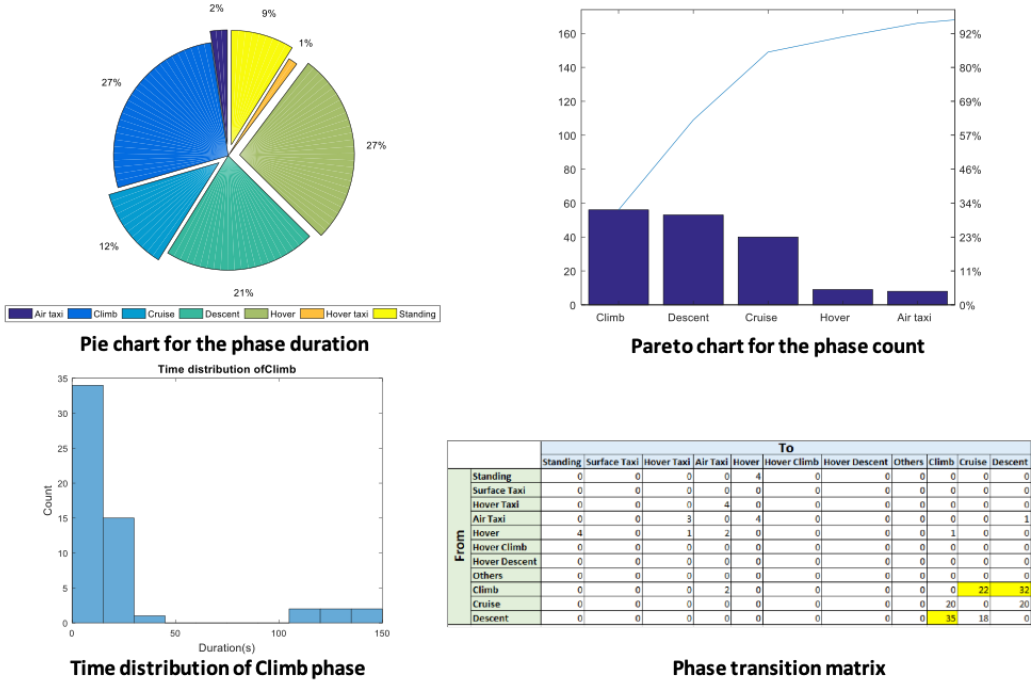


Figure 5.2: An implementation of the dashbord for displaying flight phase information

Regarding the work on anomaly detection for flight data records, some future efforts worth pursuing as the continuation of the current work are recommended as follows:

- To consult with the operators or the SMEs for verifying the potential anomalies found using our current approach. With the feedback, the algorithm developed can be tuned for better addressing the need from the relevant parties.

- To develop an automated process for handling the situation without any anomalies in the flight data records.

- To explore other types of autoencoders, such as the one with long short-term memory (LSTM) units.

For the exploratory analysis on a hazardous event, the following items are suggested for future study:

- To enhance the model fidelity of the helicopter from a two-dimensional model in a vertical plane to a three-dimensional rigid body. For example, we could potentially use a commercially available software like FlightLab to better represent the dynamics of a real helicopter.

- To apply this framework on other hazardous events, such as the encounter of vortex ring state (VRS), and the event of loss of tail rotor effectiveness (LTE).

- The current control trajectories acquired from the surrogate belong to the open-loop control strategy. They cannot be applied in real-time to a vehicle in an environment with disturbance. To achieve closed-loop control, the computed controls from the surrogate may serve as a baseline trajectory for the controller development.

Appendices

APPENDIX A

ALGORITHMS

A.1 Piecewise linear regression

Algorithm 1: Piecewise Linear Regression (PLR)

Input: Timestamp $[t_i]$, altitude $[h_i]$, where $i = 1 \cdots n$
Output: Labels of flight phases $[label_i]$
Parameter : $R^2_{threshold}$, $S_{threshold}$, $windowSize$

/* Initialization */
$\boldsymbol{x} = []$, $\boldsymbol{y} = []$, $S = []$, $segLength = []$, $label = []$, $count = 0$
/* Perform the segmentation */
for $i \leftarrow 1$ **to** n **do**
  $\boldsymbol{x} = \boldsymbol{x}.append(t_i)$
  $\boldsymbol{y} = \boldsymbol{y}.append(h_i)$
  if $length(\boldsymbol{x}) > windowSize$ **then**
    $[R^2, \sim] =$ `LinearRegression(`$\boldsymbol{x}, \boldsymbol{y}$`)`
    if $R^2 < R^2_{threshold}$ **then**
      $[\sim, slope] =$ `LinearRegression(`$\boldsymbol{x}[:-1], \boldsymbol{y}[:-1]$`)`
      $S = S.append(slope)$
      $segLength = segLength.append(length(\boldsymbol{x}) - 1)$
      $\boldsymbol{x} = \boldsymbol{x}[-1]$
      $\boldsymbol{y} = \boldsymbol{y}[-1]$
      $count$ += 1
/* Assign the flight phases */
for $j \leftarrow 1$ **to** $count$ **do**
  if $S[j] > S_{threshold}$ **then**
    for $k \leftarrow 1$ **to** $segLength[j]$ **do**
      $label = label.append$('Climb')
  else if $S[j] < -S_{threshold}$ **then**
    for $k \leftarrow 1$ **to** $segLength[j]$ **do**
      $label = label.append$('Descent')
  else
    for $k \leftarrow 1$ **to** $segLength[j]$ **do**
      $label = label.append$('Cruise')
return $label$

/* This is a function used to compute the R^2 and the slope of the regression line */
Function `LinearRegression(`$\boldsymbol{x}, \boldsymbol{y}$`)`**:**
  $R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$
  $slope = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$
  return R^2, $slope$

A.2 Sliding window regression classification

Algorithm 2: Sliding Window Regression Classification (SWRC)

Input: Timestamp $[t_i]$, altitude $[h_i]$, where $i = 1 \cdots n$
Output: Labels of flight phases $[label_i]$
Parameter : $S_{threshold}$, $windowSize$

/* Initialization */
$windowSize = k$
$labelPool \in R^{(n-k+1)\times k}$ // a matrix to store results
/* Sliding window regression implementation */
for $i \leftarrow 1$ **to** $(n-k+1)$ **do**
    $X_{reg} = \boldsymbol{t}[i : i+k-1]$
    $Y_{reg} = \boldsymbol{h}[i : i+k-1]$
    $slope =$ `LinearRegression`(X_{reg}, Y_{reg})
    if $slope > S_{threshold}$ **then**
        $labelPool[i, i : i+k-1] =$ 'Climb'
    else if $slope < -S_{threshold}$ **then**
        $labelPool[i, i : i+k-1] =$ 'Descent'
    else
        $labelPool[i, i : i+k-1] =$ 'Cruise'
/* Classification by the majority */
for $i \leftarrow 1$ **to** n **do**
    if `mode`$(labelPool[:, i])$ `=` 'Climb' **then**
        $label_i =$ 'Climb'
    else if `mode`$(labelPool[:, i])$ `=` 'Descent' **then**
        $label_i =$ 'Descent'
    else
        $label_i =$ 'Cruise'
    return $\boldsymbol{label}$

/* This is a function used to compute the slope for the regression line */
Function `LinearRegression`$(\boldsymbol{x}, \boldsymbol{y})$**:**
    $slope = \frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sum(x_i-\bar{x})^2}$
    return $slope$

A.3 Sequence smoother

Algorithm 3: Sequence smoother

Input: A sequence of flight phases $\{A_i\}$ which contains both long and short duration segments, where $i = 1, \cdots, m$

Output: An updated sequence of flight phases $\{A'_j\}$ which contains only long duration segments, where $j = 1, \cdots, n$, $n \leq m$

Parameter : *minDuration*

/* Scenario 1 */

if `isShort`$(A_1) \wedge \neg$ `isShort`(A_2) **then**
  replace the label of A_1 with the label of A_2
  combine A_1 and A_2 into a new A_1 and update the index m

/* Scenario 2 */

if $\neg$`isShort`$(A_{m-1}) \wedge$ `isShort`(A_m) **then**
  replace the label of A_m with the label of A_{m-1}
  combine A_m and A_{m-1} into a new A_m and update the index m

/* Scenario 3 */

for $i \leftarrow 1$ **to** $m - 2$ **do**
  if $\neg$`isShort`$(A_i) \wedge$ `isShort`$(A_{i+1}) \wedge \neg$`isShort`(A_{i+2}) **then**
    replace the label of A_{i+1} with the label of A_i
    combine A_i and A_{i+1} into a new A_i and update the index m

/* Scenario 4 */

$\forall$ subsequences $\{A_j, \cdots, A_k\}$, where $1 \leq j < k \leq m$
if $\neg$ `isShort`$(A_j) \wedge$ `isShort`$(\{A_{j+1}, \cdots, A_{k-1}\}) \wedge \neg$ `isShort`(A_k) **then**
  if `isShort`$(\{A_j, \cdots, A_k\})$ **then**
    replace the label of $\{A_{j+1}, \cdots, A_{k-1}\}$ with the label of A_j
    combine $\{A_j, \cdots, A_{k-1}\}$ into a new A_j and update the index m
  else
    perform linear regression on altitude data within the duration from A_{j-1} to A_{k-1}; use the slope of the regression line to update the phase labels in $\{A_{j+1}, \cdots, A_{k-1}\}$

return A_i

Function `isShort`(*Segment*)**:**
  return $len(Segment) < minDuration$

A.4 Takeoff identification

Algorithm 4: Takeoff phase identification

Input: $FlightData$, flight phase labels $[label_i]$ where $i = 1, \cdots, n$
Output: $label[j:k] \in$ types of takeoff considered, where $1 < j < k < n$

/* extract relevant variables */
$t = FlightData.time$
$WoW = FlightData.weightOnWheels$
$alt = FlightData.altitude$
$gs = FlightData.groundSpeed$
$fpa = FlightData.flightPathAngle$

/* estimate the liftoff point */
if $WoW \neq \emptyset$ **then**
    t_l is the timestamp for the liftoff where l is the first index for the variable WoW to switch to 0
else
    t_l is the timestamp for the liftoff where l is the first index for the variable alt to reach above 2 feet above ground level

/* end of the takeoff */
t_k is the timestamp where the alt variable reaches above 150 feet and $l < k < n$

/* detect rolling takeoff */
$t_m = t_k - 5\,\text{sec}$
if $gs[t_m : t_k] > 10\ knots$ **then**
    find the first index j for $gs(t_j) > 0\ knots,\ j < m$
    $label[j:k] =$ 'Rolling Takeoff'

/* detect takeoff from hover */
$j \leftarrow l$
if $\exists$'Hover' $\in label[j:k] \wedge t[label =$ 'Hover'$] > 5\,\text{sec}$ **then**
    $label[j:k] =$ 'Takeoff from Hover'

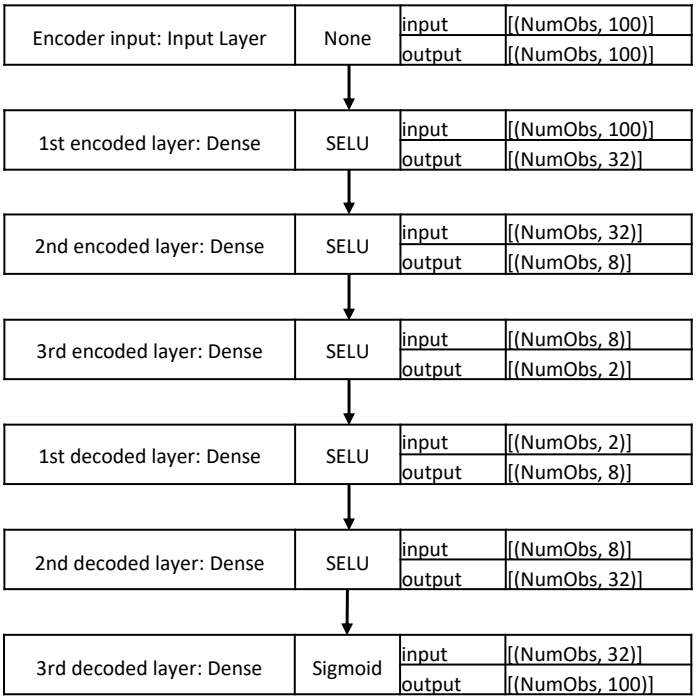
/* detect normal and maximum performance takeoff using a classifer */
$\vec{x}_{feature} = [x_1, x_2] = [max(fpa), mean(fpa)]$
$label[j:k] =$ `SVMclassifier(`$\vec{x}_{feature}$`)`
return $label[j:k]$

Function `SVMclassifier(`$\vec{x}_{feature}$`):`
    if $\vec{w} \cdot \vec{x}_{feature} + b \geq 1$ **then**
        $result =$ 'Max Performance Takeoff'
    else
        $result =$ 'Normal Takeoff'
    return $result$

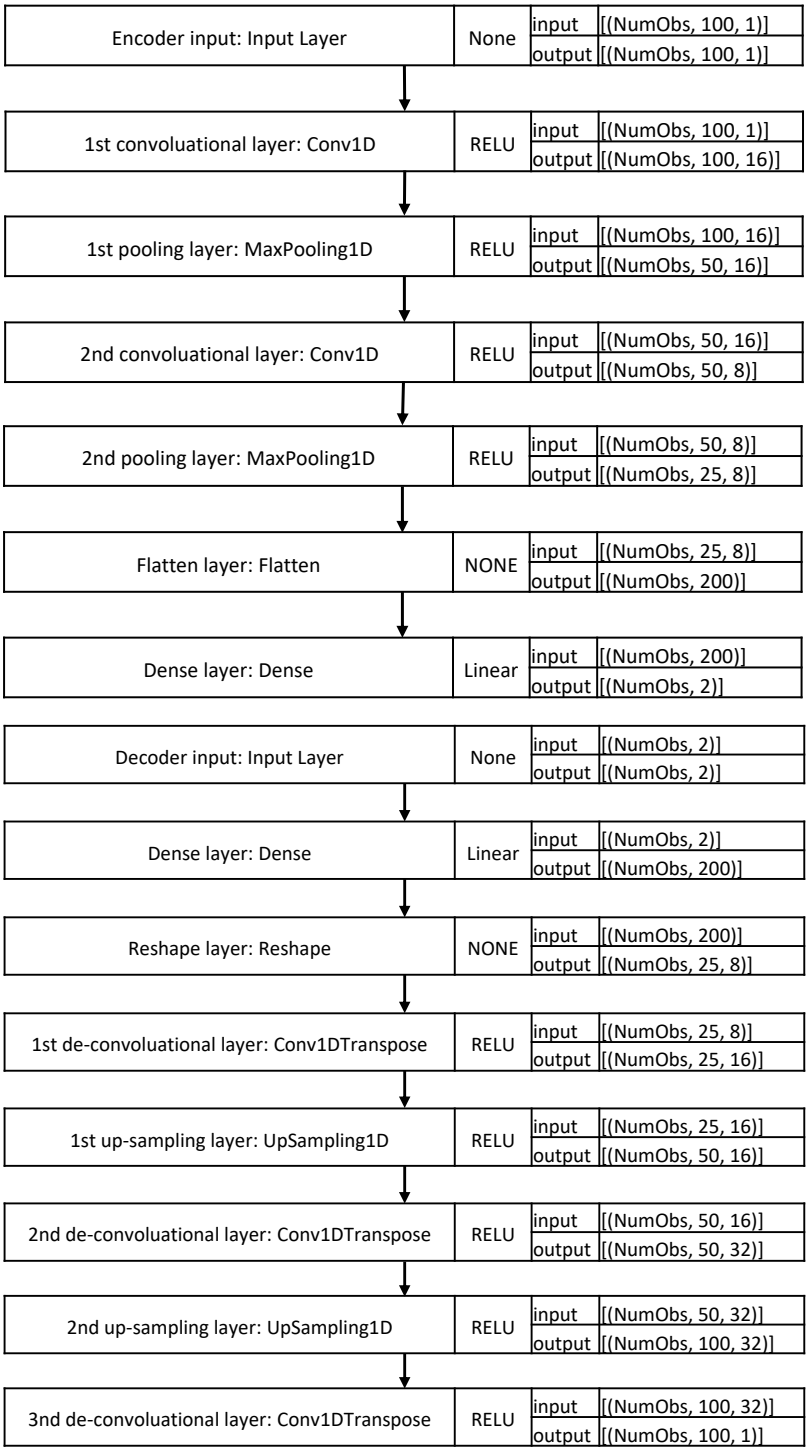
APPENDIX B

LAYOUT OF DIFFERENT AUTOENCODERS USED IN THE STUDY

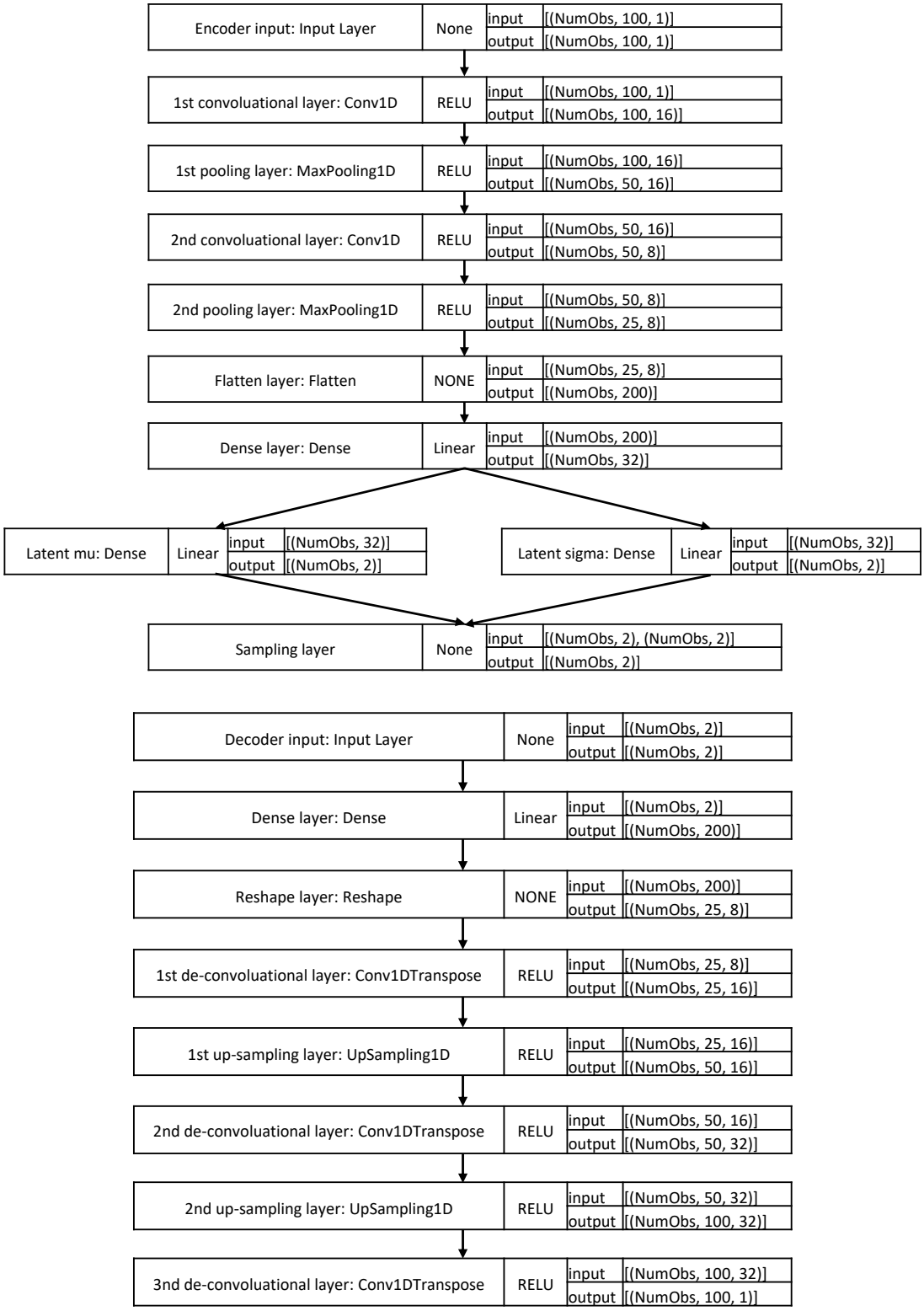
B.1 Nonlinear stacked autoencoder



B.2 One-dimensional convolutional autoencoder



B.3 Convolutional variational autoencoder



APPENDIX C

FUNCTIONS USED IN THE TESTS

In appendix C, the functions selected from [86] [87][88] and are used for testing the surrogate models and the metrics proposed for determining the number of runs. They are documented as follows.

C.1 Franke's function

$$f(\boldsymbol{x}) = 0.75\exp\left[-\frac{(9x_1-2)^2}{4} - \frac{(9x_2-2)^2}{4}\right] + 0.75\exp\left[-\frac{(9x_1+2)^2}{49} - \frac{9x_2+1}{10}\right] + 0.5\exp\left[-\frac{(9x_1-7)^2}{4} - \frac{(9x_2-3)^2}{4}\right] - 0.2\exp\left[-(9x_1-4)^2 - (9x_2-7)^2\right]$$

where $x_i \in [0,1]$, for $i = 1,2$

C.2 Mishra's bird function

$$f(\boldsymbol{x}) = \sin(x_2)\exp\left[(1-\cos(x_1))^2\right] + \cos(x_1)\exp\left[(1-\sin(x_2))^2\right] + (x_1-x_2)^2$$

where

$$x_1 \in [-10, 0]$$

$$x_2 \in [-6.5, 0]$$

C.3 Townsend's function

$$f(\boldsymbol{x}) = -[\cos((x_1-0.1)x_2)]^2 - x_1\sin(3x_1+x_2)$$

where

$$x_1 \in [-2.25, 2.25]$$

$$x_2 \in [-2.5, 1.75]$$

C.4 OTL circuit function

$$f(\boldsymbol{x}) = \frac{(V_{b1} + 0.74)x_6(x_5 + 9)}{x_6(x_5 + 9) + x_3} + \frac{11.35x_3}{x_6(x_5 + 9) + x_3} + \frac{0.74x_3x_6(x_5 + 9)}{(x_6(x_5 + 9) + x_3)x_4}$$

where

$$V_{b1} = 12x_1/(x_1 + x_2)$$

$$x_1 \in [50, 150]$$

$$x_2 \in [25, 70]$$

$$x_3 \in [0.5, 3]$$

$$x_4 \in [1.2, 2.5]$$

$$x_5 \in [0.25, 1.2]$$

$$x_6 \in [50, 300]$$

C.5 Piston function

$$f(\boldsymbol{x}) = 2\pi \sqrt{\frac{x_1}{x_4 + x_2^2 \frac{x_3 x_5}{x_7} \frac{x_6}{V^2}}}$$

where

$$V = \frac{x_2}{2x_4}\left(\sqrt{A^2 + 4x_4\frac{x_3 x_5}{x_7}x_6} - A\right)$$

$$A = x_2 x_5 + 19.62x_1 - \frac{x_3 x_4}{x_2}$$

$$x_1 \in [30, 60]$$

$$x_2 \in [0.005, 0.02]$$

$$x_3 \in [0.002, 0.01]$$

$$x_4 \in [1000, 5000]$$

$$x_5 \in [90000, 110000]$$

$$x_6 \in [290, 296]$$

$$x_7 \in [340, 360]$$

C.6 Borehole function

$$f(\boldsymbol{x}) = \frac{2\pi x_3(x_4 - x_6)}{\ln(x_2/x_1)\left(1 + \frac{2x_3 x_7}{\ln(x_2/x_1)x_1^2 x_8} + \frac{x_3}{x_5}\right)}$$

where

$$x_1 \in [0.05, 0.15]$$

$$x_2 \in [100, 50000]$$

$$x_3 \in [63070, 115600]$$

$$x_4 \in [990, 1110]$$

$$x_5 \in [63.1, 116]$$

$$x_6 \in [700, 820]$$

$$x_7 \in [1120, 1680]$$

$$x_8 \in [985, 12045]$$

C.7 Robot arm function

$$f(\boldsymbol{x}) = \sqrt{(u^2 + v^2)}$$

where

$$u = \sum_{i=1}^{4} x_i \cos\left(\sum_{j=5}^{i+4} x_j\right)$$
$$v == \sum_{i=1}^{4} x_i \sin\left(\sum_{j=5}^{i+4} x_j\right)$$

where

$$x_1 \in [0, 1]$$

$$x_2 \in [0, 1]$$

$$x_3 \in [0, 1]$$

$$x_4 \in [0, 1]$$

$$x_5 \in [0, 2\pi]$$

$$x_6 \in [0, 2\pi]$$

$$x_7 \in [0, 2\pi]$$

$$x_8 \in [0, 2\pi]$$

REFERENCES

[1] N. S. Levi, *Can helicopter flight-seeing tours be safe?* https://www.travelersunited.org/can-helicopter-flight-seeing-tours-safe/, 2018.

[2] United States Helicopter Safety Team, *Monthly safety report*, https://www.ushst.org/Reports/USHST_SafetyReport_20191108.pdf, 2019.

[3] A. H. Rao and K. Marais, "Comparing hazardous states and trigger events in fatal and non-fatal helicopter accidents," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3916.

[4] U.S. Joint Helicopter Safety Analysis Team, *The Compendium Report: The U.S. JHSAT Baseline of Helicopter Accident Analysis*, http://www.ihst.org/portals/54/US_JSHAT_Compendium_Report1.pdf, 2011.

[5] S. K. Cusick, A. I. Cortes, and C. C. Rodrigues, *Commercial aviation safety*. McGraw Hill Professional, 2017.

[6] Federal Aviation Administration, *Safety management system components*, https://www.faa.gov/about/initiatives/sms/explained/components/.

[7] Civil Aviation Authority, "Cap 739: Flight data monitoring," 2013.

[8] A. P. Payan, A. Gavrilovski, H. Jimenez, and D. N. Mavris, "Review of proactive safety metrics for rotorcraft operations and improvements using model-based parameter synthesis and data fusion," in *AIAA Infotech@ Aerospace*, 2016, p. 2133.

[9] R. G. Fox, "The history of helicopter safety," in *International helicopter safety symposium*, 2005, pp. 1–17.

[10] L. Iseler, J. DeMaio, and M. Rutkowski, "An analysis of us civil rotorcraft accidents by cost and injury (1990-1996)," 2002.

[11] U.S. Helicopter Safety Team, *U.s. helicopter safety team warns about the next "bump in the road" for fatal accidents*, http://ushst.org.dnn4less.net/Portals/0/2019-Release-Oct-and-Nov.pdf, 2019.

[12] NTSB, *NTSB 2017-2018 Most wanted list of transportation safety improvements*, https://www.ntsb.gov/safety/mwl/Documents/2017-18/2017MWL-FctSht-Recorders-A.pdf.

[13] A. Gavrilovski *et al.*, "Challenges and opportunities in flight data mining: A review of the state of the art," in *AIAA Infotech@ Aerospace*, 2016, p. 0923.

[14] A. P. Payan, P.-N. Lin, C. Johnson, and D. N. Mavris, "Helicopter approach stability analysis using flight data records," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3437.

[15] A. P. Payan, A. Gavrilovski, H. Jimenez, and D. N. Mavris, "Improvement of rotorcraft safety metrics using performance models and data integration," *Journal of Aerospace Information Systems*, pp. 26–39, 2017.

[16] Federal Aviation Administration, *Helicopter flying handbook*, https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/helicopter_flying_handbook/, 2019.

[17] Bell Helicopter, *Bell model 206b rotorcraft flight manual*, https://www.mvheli.com/wp-content/uploads/bell-206b3-fm-1.pdf, 2000.

[18] B. G. Amidan and T. A. Ferryman, "Atypical event and typical pattern detection within complex systems," in *2005 IEEE Aerospace Conference*, IEEE, 2005, pp. 3620–3631.

[19] D. L. Iverson, "Inductive system health monitoring with statistical metrics," 2005.

[20] 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)*, vol. 39, no. 1, pp. 101–113, 2008.

[21] S. Das, B. L. Matthews, A. N. Srivastava, and N. C. Oza, "Multiple kernel learning for heterogeneous anomaly detection: Algorithm and aviation safety case study," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 47–56.

[22] E. Chu, D. Gorinevsky, and S. Boyd, "Detecting aircraft performance anomalies from cruise flight data," in *AIAA Infotech@ Aerospace 2010*, 2010, p. 3307.

[23] S. Das, S. Sarkar, A. Ray, A. Srivastava, and D. L. Simon, "Anomaly detection in flight recorder data: A dynamic data-driven approach," in *2013 American Control Conference*, IEEE, 2013, pp. 2668–2673.

[24] K. Bhaduri, B. L. Matthews, and C. R. Giannella, "Algorithms for speeding up distance-based outlier detection," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 859–867.

[25] S. D. Bay and M. Schwabacher, "Mining distance-based outliers in near linear time with randomization and a simple pruning rule," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 29–38.

[26] C. Rao, A. Ray, S. Sarkar, and M. Yasar, "Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns," *Signal, Image and Video Processing*, vol. 3, no. 2, pp. 101–114, 2009.

[27] L. Li, S. Das, R John Hansman, R. Palacios, and A. N. Srivastava, "Analysis of flight data using clustering techniques for detecting abnormal operations," *Journal of Aerospace information systems*, vol. 12, no. 9, pp. 587–598, 2015.

[28] S. Das, L. Li, A. Srivastava, and R. J. Hansman, "Comparison of algorithms for anomaly detection in flight recorder data of airline operations," in *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2012, p. 5593.

[29] L. Li, R. J. Hansman, R. Palacios, and R. Welsch, "Anomaly detection via a gaussian mixture model for flight operation and safety monitoring," *Transportation Research Part C: Emerging Technologies*, vol. 64, pp. 45–57, 2016.

[30] T. G. Puranik, H. Jimenez, and D. N. Mavris, "Utilizing energy metrics and clustering techniques to identify anomalous general aviation operations," in *AIAA Information Systems-AIAA Infotech@ Aerospace*, 2017, p. 0789.

[31] R. Deshmukh and I. Hwang, "Anomaly detection using temporal logic based learning for terminal airspace operations," in *AIAA Scitech 2019 Forum*, 2019, p. 0682.

[32] G. Jarry, D. Delahaye, F. Nicol, and E. Feron, "Aircraft atypical approach detection using functional principal component analysis," *Journal of Air Transport Management*, vol. 84, p. 101 787, 2020.

[33] M. Memarzadeh, B. Matthews, and I. Avrekh, "Unsupervised anomaly detection in flight data using convolutional variational auto-encoder," *Aerospace*, vol. 7, no. 8, p. 115, 2020.

[34] L. Basora, X. Olive, and T. Dubot, "Recent advances in anomaly detection methods applied to aviation," *Aerospace*, vol. 6, no. 11, p. 117, 2019.

[35] T. W. Liao, "Clustering of time series data—a survey," *Pattern recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.

[36] S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah, "Time-series clustering–a decade review," *Information Systems*, vol. 53, pp. 16–38, 2015.

[37] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly detection in time series data," *arXiv preprint arXiv:2002.04236*, 2020.

[38] Z. Zhang, K. Huang, and T. Tan, "Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes," in *18th International Conference on Pattern Recognition (ICPR'06)*, IEEE, vol. 3, 2006, pp. 1135–1138.

[39] Y. Zheng, "Trajectory data mining: An overview," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, pp. 1–41, 2015.

[40] H. Su, S. Liu, B. Zheng, X. Zhou, and K. Zheng, "A survey of trajectory distance measures and performance evaluation," *The VLDB Journal*, vol. 29, no. 1, pp. 3–32, 2020.

[41] W. Johnson, "Helicopter optimal descent and landing after power loss," 1977.

[42] A. Y. Lee, A. E. Bryson, and W. S. Hindson, "Optimal landing of a helicopter in autorotation," *Journal of Guidance, Control, and Dynamics*, vol. 11, no. 1, pp. 7–12, 1988.

[43] Y. Okuno, K. Kawachi, A. Azuma, and S. Saito, "Analytical prediction of height-velocity diagram of a helicopter using optimal control theory," *Journal of Guidance, Control, and Dynamics*, vol. 14, no. 2, pp. 453–459, 1991.

[44] Y. Zhao, A. A. Jhemi, and R. T. Chen, "Optimal vertical takeoff and landing helicopter operation in one engine failure," *Journal of Aircraft*, vol. 33, no. 2, pp. 337–346, 1996.

[45] E. B. Carlson and Y. J. Zhao, "Prediction of tiltrotor height-velocity diagrams using optimal control theory," *Journal of aircraft*, vol. 40, no. 5, pp. 896–905, 2003.

[46] J. T. Betts, *Practical methods for optimal control and estimation using nonlinear programming*. Siam, 2010, vol. 19.

[47] E. N. Bachelder and B. L. Aponso, "An autorotation flight director for helicopter training," in *ANNUAL FORUM PROCEEDINGS-AMERICAN HELICOPTER SOCIETY*, AMERICAN HELICOPTER SOCIETY, INC, vol. 59, 2003, pp. 1861–1872.

[48] A. A. Jhemi, E. B. Carlson, Y. J. Zhao, and R. T. Chen, "Optimization of rotorcraft flight following engine failure," *Journal of the American Helicopter Society*, vol. 49, no. 2, pp. 117–126, 2004.

[49] E. Carlson, S Xue, J Keane, and B Kevin, "H-1 upgrades height-velocity diagram development through flight test and trajectory optimization," in *ANNUAL FORUM*

PROCEEDINGS-AMERICAN HELICOPTER SOCIETY, AMERICAN HELICOPTER SOCIETY, INC, vol. 62, 2006, p. 729.

[50] P. Abbeel, A. Coates, T. Hunter, and A. Y. Ng, "Autonomous autorotation of an rc helicopter," in *Experimental Robotics*, Springer, 2009, pp. 385–394.

[51] D. J. Lee, H. Bang, and K. Baek, "Autorotation of an unmanned helicopter by a reinforcement learning algorithm," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 7279.

[52] D. J. Lee and H. Bang, "Autonomous autorotation of an unmanned helicopter using a reinforcement learning algorithm," *Journal of Aerospace Information Systems*, vol. 10, no. 2, pp. 98–104, 2013.

[53] S. Tierney, "Autorotation path planning using backwards reachable set and optimal control," 2010.

[54] T. Yomchinda, J. Horn, and J. Langelaan, "Flight path planning for descent-phase helicopter autorotation," in *AIAA Guidance, Navigation, and Control Conference*, 2011, p. 6601.

[55] N. A. Grande, "Safe autonomous flare and landing during autorotation through wind shear," 2013.

[56] N. Grande, S. Tierney, J. F. Horn, and J. W. Langelaan, "Safe autorotation through wind shear via backward reachable sets," *Journal of the American Helicopter Society*, vol. 61, no. 2, pp. 1–11, 2016.

[57] K. Dalamagkidis, K. P. Valavanis, and L. A. Piegl, "Autonomous autorotation of unmanned rotorcraft using nonlinear model predictive control," in *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*, Springer, 2009, pp. 351–369.

[58] P. Bibik and J. Narkiewicz, "Helicopter optimal control after power failure using comprehensive dynamic model," *Journal of guidance, control, and dynamics*, vol. 35, no. 4, pp. 1354–1362, 2012.

[59] Z. Sunberg and J. Rogers, "A fuzzy logic-based controller for helicopter autorotation," in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 2013, p. 1150.

[60] B. Eberle, J. Rogers, M. Jump, and N. Cameron, "Time-to-contact-based control laws for flare trajectory generation and landing point tracking in autorotation," 2018.

[61] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker, "Surrogate-based analysis and optimization," *Progress in aerospace sciences*, vol. 41, no. 1, pp. 1–28, 2005.

[62] V. R. Joseph, E. Gul, and S. Ba, "Maximum projection designs for computer experiments," *Biometrika*, vol. 102, no. 2, pp. 371–380, 2015.

[63] V. R. Joseph, "Space-filling designs for computer experiments: A review," *Quality Engineering*, vol. 28, no. 1, pp. 28–35, 2016.

[64] Y. Hung, V. R. Joseph, and S. N. Melkote, "Analysis of computer experiments with functional response," *Technometrics*, vol. 57, no. 1, pp. 35–44, 2015.

[65] E. Gul, V. R. Joseph, H. Yan, and S. N. Melkote, "Uncertainty quantification of machining simulations using an in situ emulator," *Journal of Quality Technology*, vol. 50, no. 3, pp. 253–261, 2018.

[66] S. Mak *et al.*, "An efficient surrogate model for emulation and physics extraction of large eddy simulations," *Journal of the American Statistical Association*, vol. 113, no. 524, pp. 1443–1456, 2018.

[67] F. A. Administration, *Federal aviation regulations/aeronautical information manual*, https://www.faa.gov/air_traffic/publications/atpubs/aim_html/index.html.

[68] I. C. T. Team, *Phase of flight - definitions and usage notes*, https://www.ntsb.gov/investigations/data/Documents/datafiles/PhaseofFlightDefinitions.pdf.

[69] E. Aviation, *Data definition standard*, https://www.icao.int/safety/airnavigation/AIG/Documents/ADREP%20Taxonomy/ECCAIRS%20Aviation%201.3.0.12%20(Entities%20and%20Attributes).en.id.pdf.

[70] V. Goblet, N. Fala, and K. Marais, "Identifying phases of flight in general aviation operations," in *15th AIAA Aviation Technology, Integration, and Operations Conference*, 2015, p. 2851.

[71] W. E. Kelly and J. H. Painter, "Flight segment identification as a basis for pilot advisory systems," *Journal of aircraft*, vol. 43, no. 6, pp. 1628–1635, 2006.

[72] J. N. Robinson, A. P. Payan, C. Johnson, and D. N. Mavris, "Visual and instrument helicopter approach stability analysis using data fusion and data analytics," in *AIAA Scitech 2021 Forum*, 2021, p. 0528.

[73] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in *Data mining in time series databases*, World Scientific, 2004, pp. 1–21.

[74] M. Lovrić, M. Milanović, and M. Stamenković, "Algoritmic methods for segmentation of time series: An overview," *Journal of Contemporary Economic and Business Issues*, vol. 1, no. 1, pp. 31–53, 2014.

[75] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[76] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[77] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *arXiv preprint arXiv:1906.02691*, 2019.

[78] M. Abadi *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015.

[79] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*, IEEE, 2008, pp. 413–422.

[80] M. P. Kelly, *Optimtraj user's guide, version 1.5*, https://github.com/MatthewPeterKelly/OptimTraj, 2019.

[81] M. McKay, R. Beckman, and W. Conover, "Comparison the three methods for selecting values of input variable in the analysis of output from a computer code," *Technometrics;(United States)*, vol. 21, no. 2, 1979.

[82] B. Tang, "Orthogonal array-based latin hypercubes," *Journal of the American statistical association*, vol. 88, no. 424, pp. 1392–1397, 1993.

[83] M. E. Johnson, L. M. Moore, and D. Ylvisaker, "Minimax and maximin distance designs," *Journal of statistical planning and inference*, vol. 26, no. 2, pp. 131–148, 1990.

[84] M. D. Morris and T. J. Mitchell, "Exploratory designs for computational experiments," *Journal of statistical planning and inference*, vol. 43, no. 3, pp. 381–402, 1995.

[85] I. M. Sobol', "On sensitivity estimation for nonlinear mathematical models," *Matematicheskoe modelirovanie*, vol. 2, no. 1, pp. 112–118, 1990.

[86] S. Surjanovic and D. Bingham, *Virtual library of simulation experiments: Test functions and datasets*, Retrieved October 25, 2021, from http://www.sfu.ca/~ssurjano.

[87] S. K. Mishra, "Some new test functions for global optimization and performance of repulsive particle swarm method," *Available at SSRN 926132*, 2006.

[88] A Townsend, "Constrained optimization in chebfun," *Chebfun. org*, 2014.

[89] J. L. Loeppky, J. Sacks, and W. J. Welch, "Choosing the sample size of a computer experiment: A practical guide," *Technometrics*, vol. 51, no. 4, pp. 366–376, 2009.