Online Prediction of Mid-Flight Aircraft Trajectories with Multi-Timestep Markov Models

by

Yongzhen Arthur Pan

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Online trajectory prediction is central to the function of air traffic control of improving the flow of air traffic and preventing collisions, particularly considering the ever-increasing number of air travellers. In this thesis, we propose an approach to predict the mid-flight trajectory of an aircraft using models learned from historical trajectories. The main idea is based on Markov Models with transition probabilities for multiple timesteps, representing the location of aircraft as states and movement of aircraft over a certain number of minutes (*i.e.* timestep) as transition probabilities between states. Using our approach, one is able to make predictions of future positions of mid-flight aircraft for each minute up to twenty minutes into the future, and concatenate them to form the predicted trajectory of the aircraft for the next twenty minutes. We evaluated the effectiveness of the proposed approach using a dataset of historical trajectories over the USA. Using prediction accuracy metrics from the aviation domain, we demonstrated that our approach was able to make accurate predictions of future trajectories of mid-flight aircraft, achieving an improvement of 24.6% in horizontal error and 34.2% in vertical error over baseline models from conventional approaches, with each prediction requiring mere milliseconds to compute.

Preface

This thesis is an original work by Yongzhen Arthur Pan, completed under the guidance of Professor Mario A. Nascimento and Professor Joerg Sander. Chapter 3 and Sections 4.1, 5.1, & 5.3 have been published [34]. The remaining material are either new or have gone through major revisions, and may be submitted for future publications.

Acknowledgements

The work presented in this thesis would not have been possible without the encouragement and support of the people around me. I would like to take this opportunity to express my gratitude and appreciation to all who have made this thesis possible.

First and foremost, I am deeply grateful to Professor Mario A. Nascimento and Professor Joerg Sander, for accepting me as a student and for their tireless guidance throughout my research. They have opened the doors for me toward a world of possibilities and opportunities. I would also like to thank Professor Davood Rafiei for serving on my examining committee.

I am indebted to DSO National Laboratories (Singapore) for sponsoring my graduate studies, and to Dr. How Khee Yin for his support in my application. The sponsorship allayed any financial stress that I might face from pursuing graduate education while having to support my family at the same time. It also helped ensure that my research remain grounded in tackling the problems faced by my country and my people. I would also like to thank my various colleagues who have given me their help and advice at different times in my career.

I am blessed with loving family and friends from a plethora of countries and cultures; some of whom have made every gathering in Canada feel like a festival, some of whom eagerly look forward to my next visit to Japan, and some of whom longingly await my return home to Singapore.

Finally, I am thankful for having a devoted wife and a loving sister who have stood by me all these years.

Contents

List of Tables

List of Figures

List of Abbreviations

IATA	International Air Transport Association
NextGen	Next Generation Air Transportation System
FAA	Federal Aviation Administration
NAS	National Airspace System
SESAR	Single European Sky ATM Research
ATM	Air Traffic Management
ASM	Airspace Management
ATS	Air Traffic Services
ATFM	Air Traffic Flow Management
ATC	Air Traffic Control
TP	Trajectory Prediction
HMM	Hidden Markov Model
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
PBN	Performance-Based Navigation
TBO	Trajectory-Based Operations
PCA	Principal Component Analysis
STAR	Standard Terminal Arrival Route
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise
MM	Markov Model
RNN	Recurrent Neural Network
ADS-B	Automatic Dependent Surveillance-Broadcast
WGS84	World Geodetic System 84
RAP	Rapid Refresh weather forecasting system

NOAA	National Oceanic and Atmospheric Administration
CONUS	Continental United States
LGA	IATA code for LaGuardia Airport
ORD	IATA code for Chicago O'Hare Airport
BOS	IATA code for Boston Logan Airport
KBM	Kinematics-Based Model
MTM	Median Trajectory Model
DTW	Dynamic Time Warping
OMM	Observed Markov Model

List of Symbols

gp	A geographical position consisting of latitude, longitude, and altitude
ts	A timestamp
pos	A positional update of an aircraft, consisting of a geographical position and a timestamp
Trj	A raw trajectory of an aircraft, consisting of a finite sequence of positional updates
$\overline{Trj}$	An aligned trajectory
rp	A reference position in our 3-dimensional reference grid
gc	A grid cube centered upon a reference position
$ref(\cdot)$	Function that maps a grid cube to its reference point, *e.g.* $ref(gc) = rp$
G	The set of all grid cubes
wv	A weather vector at a particular time in a particular grid cube consisting specific humidity, temperature, wind speed, and wind direction
sh	Specific humidity
tk	Temperature (in Kelvins)
ws	Wind speed
wd	Wind direction
wb	A discrete bin vector mapped from a weather vector
$bin(\cdot)$	Function that maps a weather vector to its bin vector, *e.g.* $bin(wv) = wb$
W	The set of all possible bin vectors
N	The number of possible states in a MM (Markov Model)
S	The set of all possible states in a MM; set elements have a one-to-one correspondence with those in set G
s_i or s_j or s_l	A particular state in a MM, *i.e.* $s_i \in S$ or $s_j \in S$ or $s_l \in S$
$a_t(\cdot, \cdot)$	The transition probability from one state to another state in a MM after t minutes, *e.g.* $a_t(s_i, s_j)$; the time taken for the transition, t minutes, is also called a 'timestep'
M	The number of possible observations in an HMM (Hidden Markov Model)
U	The set of all possible observations in an HMM; set elements have a one-to-one correspondence with those in set W

u_k	A particular observation in an HMM, *i.e.* $u_k \in U$
$b(\cdot, \cdot)$	The emission probability of an observation at a state in an HMM, *e.g.* $b(s_j, u_k)$
L	The look-ahead time, which is set at 20 minutes for this thesis, *i.e.* future positions of an aircraft are predicted for each minute into the future, up to the look-ahead time of 20 minutes
ts_{now}	The current timestamp of a particular prediction for our proposed models, *i.e.* the 'starting' timestamp that corresponds to $t = 0$ in a generic MM
$q_{ts_{now}}$	The input start state of our proposed models, belonging to the set of possible states S, *i.e.* $q_{ts_{now}} \in S$
O	An observation matrix of size $N \times L$, consisting of the observations for each of the N states over the next L minutes
$o_{j,t}$	The observation in state s_j at timestamp $ts_{now} + t$, $t \in [1, L]$, belonging to the set of possible observations U, *i.e.* $o_{j,t} \in U$
$\widehat{q}_{ts_{now}+t}$	The most probable state at timestamp $ts_{now} + t$, based on a prediction model
$vt_t(\cdot)$	The Viterbi probability for a state at timestamp $ts_{now} + t$, *e.g.* $vt_t(s_j)$
$bt_t(\cdot)$	The Viterbi backtrace for a state at timestamp $ts_{now}+t$, *e.g.* $bt_t(s_j)$
$\Gamma(\cdot)$	Function that maps a state to its neighbouring states, *e.g.* $\Gamma(s_i)$

Chapter 1

Introduction

This chapter introduces the problem we are attempting to solve, and highlights the key contributions that we have achieved. A brief outline of the remaining chapters of this thesis is also provided.

1.1 Motivation

The International Air Transport Association (IATA) reported that worldwide annual air passenger numbers reached a new height of 4.1 billion passengers in 2017, representing an increase of 7.3% from 2016, and an additional 280 million flights [17]. Based on growth projection, the IATA predicts that air passenger numbers could double to 8.2 billion by 2037 [16]. This could lead to a potential infrastructure crisis: airports and infrastructure operators will not be able to handle the demand based on the current rate of growth, hence governments and operators need to plan for infrastructure upgrades well in advance.

In order to mitigate this crisis, civil aviation authorities, air navigation service providers, and commercial airlines are participating in ongoing modernization projects that aim to improve computer systems and infrastructure used for air transportation. In the United States, a multi-decade effort called the Next Generation Air Transportation System (NextGen) project is led by the Federal Aviation Administration (FAA) in improving the overall capacity, performance, efficiency, and predictability throughout the National Airspace System (NAS) [10]. In Europe, parallel to the NextGen project, a trans-national initiative of the European

Commission called the Single European Sky ATM Research (SESAR) has the goal of improving Air Traffic Management (ATM) for the European airspace [39].

Air Traffic Management (ATM) encompasses a wide range of operational systems that assist aircraft to take-off from a departure airport, fly through an airspace, and land at an arrival airport, in a safe and efficient manner. ATM systems can be sub-divided into many areas, such as Airspace Management (ASM), Air Traffic Services (ATS), Air Traffic Flow Management (ATFM), and Air Traffic Control (ATC) [18]. In particular, Air Traffic Control (ATC) is a service provided by ground-based air traffic controllers to direct aircraft through an airspace while maintaining safe separation from other aircraft or obstacles and ensuring the safe and orderly flow of air traffic [28]. As the increase in global air traffic brings about more aircraft flying in the air at any point in time, air traffic controllers around the world face the challenge of having to direct them to safely navigate an increasingly congested airspace.

Trajectory Prediction (TP) systems are used by ATC to calculate the likely paths of aircraft currently flying in the air (*i.e.* mid-flight aircraft) over a time horizon into the future (*i.e.* look-ahead time), which aids air traffic controllers in identifying potential bottlenecks in the airspace or potential breaches in the safe separation of aircraft. Traditionally, TP systems use the current state – such as position, velocity and acceleration parameters – and intent – according to the filed flight plan – of an aircraft to extrapolate its trajectory over the next 10 to 20 minutes [15]. The prediction accuracy of such approaches may be affected by unexpected changes in aircraft intent (such as a temporary change in altitude to maintain safe separation from another aircraft), unknown aircraft specifications (based on the cargo load on the aircraft), or external factors (such as strong winds or congestion, in the airspace or at the destination airport) [15].

Recent advancements in computing research have made it possible to learn probabilistic models from historical data to be used for prediction. In this thesis, we explore this approach by building Markov Models from historical trajectory data to be used for the online prediction of mid-flight aircraft trajectories, with the aim of improving prediction accuracy in *real-time* TP systems. By providing timely pre-

dictions of mid-flight aircraft trajectories that are more accurate than predictions made using the traditional approach particularly over longer look-ahead times, our models can reduce the chances for errors made by air traffic controllers due to heavy workload or fatigue when directing a high volume of air traffic through a congested airspace. The output predictions of our models can also be used by downstream ATC applications to pin-point areas within the airspace that may become congested, or to automatically detect and resolve conflicts in the predicted trajectories that may result in mid-air collisions [2].

1.2 Contributions of this Thesis

In order to use Hidden Markov Models (HMMs) for Trajectory Prediction (TP), we need to map aircraft positions and weather observations into discrete states of the model. First, we divide the geographical volume of interest into discrete, evenly spaced grid cubes. Each hidden state represents the presence of the aircraft of interest in a corresponding grid cube, and each observation produced at a hidden state represents the aircraft experiencing a particular weather condition in the corresponding grid cube. Then, we train models from historical trajectories, based on utilizing transition probabilities of multiple timesteps from 1 to 20 minutes. The current position of the aircraft is used as the input start state for each model to predict the most likely positions of the aircraft for each one of the next 20 minutes, and the concatenation of the predicted positions form the predicted trajectory of the aircraft, in the air, for the next 20 minutes.

The contributions of this thesis to the area of Online Aircraft Trajectory Prediction can be summarized as follows:

- We mapped basic concepts for the Trajectory Prediction problem onto Hidden Markov Models that are able to incorporate local weather information, and proposed the use of Multi-Timestep Markov Models to overcome the inherent limitation of Markov Models in that there is no explicit representation of time duration in each state.
- We used more than 16,000 historical trajectories over a continuous time

span of 2 years for experimental evaluation, consisting of flights between LaGuardia Airport and Chicago O'Hare Airport (which is among the top 20 busiest domestic routes in 2017 [31]), as well as flights between Boston Logan Airport and Chicago O'Hare Airport.

- We applied relevant prediction metrics from the aviation domain [15] for a comprehensive experimental evaluation of multiple prediction models of various configurations. Our best-performing model, called the Constrained Multi-Timestep Markov Model, achieved highly accurate predictions of mid-flight aircraft trajectories, with an improvement of 24.6% in horizontal error and 34.2% in vertical error over the best-performing baseline model at the look-ahead time of 20 minutes for the busiest flight route.

1.3 Outline of this Thesis

The rest of this thesis is organized as follows. Chapter 2 gives an overview of related work, followed by some basic concept definitions and the problem definition in Chapter 3. Chapter 4 provides the theoretical basis of our proposed models, while the experimental evaluation of their effectiveness is presented in Chapter 5. Chapter 6 concludes this thesis with a brief summary and discussion of future work.

Chapter 2

Related Work

This chapter summarizes the research that has been done on aircraft trajectory data, specifically in the analysis of aircraft trajectories through clustering, and in the prediction of aircraft trajectories.

2.1 Background

The deployment of Global Navigation Satellite Systems (GNSSs), such as the Global Positioning System (GPS) and Galileo (by the European Union), has enabled the use of geo-location data for a wide variety of purposes, including navigation and orientation (*e.g.* in transportation, sports, and recreation), surveying and mapping (*e.g.* in construction, geology, and archaelogy), military operations, and even social network services [35, 42]. Furthermore, the rapid proliferation of GNSS-enabled devices, coupled with advancements in computer storage and processing capabilities, have made it possible to collect large amounts of geo-location data generated by moving objects – such as people, vehicles, animals, and natural phenomena – to be stored for further analysis. In particular, trajectory data, formed by concatenating recorded positions of a moving object in chronological order, represents the path travelled by the moving object over time, and contains important information that can provide valuable insights into the behavior and habits of a moving object [45].

Trajectory data can be broadly classified into two categories: objects moving within a land transportation network, such as cars, buses, and trains, and objects

moving in a free and open space (usually in the sky or at sea), such as aircraft, ships, and animals. The first category, trajectory data of land transport vehicles, has been widely explored in recent years because of the wide-spread adoption of map navigation applications (*e.g.* Google Maps, Waze) and ride-share services (*e.g.* Uber, Lyft). In this category, the road (or rail) network is typically represented as a graph, in which road intersections correspond to nodes and road segments correspond to edges [46]; alternatively, some approaches may use a graph where points or areas of interest correspond to nodes, while the commutes between these points or areas correspond to edges [46]. The trajectories of vehicles are then matched with nodes and edges in the graph in order to identify travel patterns, assess traffic conditions, or detect traffic anomalies [46].

The second category, trajectory data of objects moving in open spaces, can also make use of a graph-matching approach when the semantics of certain locations or regions in the open space can be exploited, such as through the use of way-points, intersections, or areas of activity in the open space. However, such semantics may not always be available, and it is subject to changes due to the dynamic nature of navigation in open spaces: unlike land transportation where the entire road or rail network that vehicles travel on have to be constructed, the path of travel in an open space is not strictly restricted to fixed paths within a network. Indeed, flight paths and procedures used to be limited by and planned according to the physical locations of ground-based navigation aids, but with the implementation of Performance-Based Navigation (PBN) and Trajectory-Based Operations (TBO) enabling point-to-point travel (in order to increase airspace capacity and fuel efficiency, in the context of NextGen and SESAR) [11, 19], aircraft are now travelling on more flexible and varied paths than ever before. Thus, the challenge that this category of trajectory data presents, is that techniques are required to process positional data drawn from a much larger, three-dimensional geographical space with little or evolving semantics, as compared to up to two dimensions for most applications for land transportation. We focus on the literature for moving object trajectory data in open spaces in the following sections.

2.2 Trajectory Data Analysis

Literature on the analysis of trajectory data typically follow a discriminative approach by identifying clusters of trajectories [6, 13, 32], sub-trajectories [3], or significant positions along trajectories [13, 32].

Gariel, Srivastava, & Feron [13] proposed two methods for finding clusters of trajectories to be used in airspace monitoring. The first method was based on clustering trajectories that have turning points at the same locations (*i.e.* waypoints), while the second method was based on clustering trajectories that have similar principal components after undergoing re-sampling, dimensionality augmentation, and PCA (Principal Component Analysis). Clustering results were validated by showing the cluster centroids (*i.e.* average trajectories) to air traffic controllers, who were able to identify all the STARs (Standard Terminal Arrival Routes) among the clusters. The clusters were then applied to real-time airspace monitoring, where the distance of the position of an aircraft to the nearest trajectory cluster was used as the anomaly score, and those with the highest anomaly scores were deemed to be deviating from typical flight paths. It was reported that for the first method, many trajectories were considered outliers as they do not fly over the discovered way-points, while for the second method, 19.5% of the trajectories were considered outliers as their principal components do not belong in a cluster. There was neither a measure for, nor an analysis of the effectiveness of using the centroids of trajectory clusters for airspace monitoring.

Basora, Morio, & Mailhot [6] and Olive & Morio [32] presented trajectory clustering methods for the analysis of air traffic flows around airports. Both methods used the Ramus-Douglas-Peucker algorithm to simplify trajectories, removing redundant positions (typically along straight parts of a trajectory) while keeping only significant positions (typically along curved parts of a trajectory). The method proposed by Basora, Morio, & Mailhot [6] for clustering trajectories was based on HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) and two distance functions between trajectories: Euclidean Distance and Symmetrized Segment-Path Distance. A centroid for each trajectory

cluster was determined by choosing the trajectory that had the minimal sum of distances to other trajectories within the cluster. For evaluation, the cluster centroids were compared qualitatively to existing flight routes, though it was reported that 32% and 25% of the trajectories (for each distance function used, respectively) were outliers, and would require further analysis. The method proposed by Olive & Morio [32] for clustering trajectories was based on clustering and sequencing their significant positions. Using the clusters of significant positions, each trajectory was converted into a sequence of significant positions. A tree was formed where each node represents a cluster of significant positions, while a parent-child relationship in the tree represents the trajectory segments flying over the child node, followed by the parent node. Each sequence of nodes from a leaf node (*i.e.* the first position of detection) in the tree to the root node (*i.e.* the airport) was considered a pattern, and the trajectories that fly over the sequence of nodes denoted by a pattern were considered one cluster. It was reported that 30% of the trajectories were outliers, consisting mostly of incomplete trajectories and trajectories with holding patterns (*e.g.* aircraft flying in a circular path due to congestion at the destination airport).

The use of trajectory clusters (and their cluster representatives) for predicting the future trajectory of a moving object can be done as follows. Based on the past trajectory of the moving object until the current point in time, first find the cluster that the moving object belongs in. Then, retrieve the cluster representative, and find the position on the representative that is nearest to the most recent position of the moving object. The trajectory of the representative from this position onwards can be considered a prediction of the future trajectory of the moving object. However, the large proportions of outliers reported in the literature aforementioned, as well as the dynamic nature of trajectory data as explained earlier, seem to suggest that clustering may not be the best approach for grouping trajectories together, because the well-separation of clusters is not guaranteed and trajectories do not fall squarely within a single cluster. Ayhan & Samet [3] took a different approach to trajectory data analysis: instead of using all trajectories of aircraft flying through a region of interest, only the trajectories of aircraft flying a

specific flight route (*i.e.* from a particular departure airport to a particular arrival airport) were used for clustering. This approach of selecting a sub-population of trajectories that fly a specific flight route removes the need to cluster trajectories of all aircraft flying in the airspace (and having most of them flying on different routes), and thus changes the approach for data analysis from a region-specific one to a route-specific one that focuses on finding the representative trajectory for a particular flight route. Positions along the input trajectories were first divided into three phases of flight: climb, en-route, and descent. k-means clustering was performed for each of the phases to obtain clusters of positions along the trajectories. The centroids of the clusters were then joined together to find the representative trajectory for the flight route. Although using only the trajectories for a specific flight route was a good approach for the problem, only one representative trajectory was generated, and the fact that different flights may take different paths from the departure airport to the arrival airport was ignored.

2.3 Trajectory Prediction (TP)

Research into the accurate prediction of aircraft trajectories has been active for at least 50 years [8]. Benoit, Storey, & Swierstra (1975) [8] outlined a family of 3 approaches for predicting the vertical component (*i.e.* altitude) during the climbing and descending phases of flight. The most basic approach assumes that the rate of climb/descent of the aircraft will remain constant, and makes the prediction by extrapolating the current position of the aircraft linearly into the future based on the recorded rate of change of altitude over time between the two latest positions of the aircraft. On top of radar positional data, the second approach incorporates the climb/descent performance data of each individual aircraft, while the third approach also includes air temperature and take-off mass into calculating the trajectory of an aircraft. Though the experimental evaluation was conducted on a wide range of 13 aircraft (most of which have now been replaced by newer models/variants), it was limited to the climb phase from 10,000 up to 30,000 feet, and ignored the horizontal dimension.

Trajectory prediction methods can be divided into nominal, worst-case, and probabilistic techniques [24, 44]. Nominal techniques, such as those described in the previous paragraph, project a single trajectory (*i.e.* a nominal trajectory) of an aircraft into the future based on its current state information (*e.g.* velocity) without considering uncertainties, but only work for very short predictions and for trajectories that are very predictable [24]. Worst-case techniques consider multiple possible maneuvers that an aircraft can make based on its capabilities, but cause a high false alarm rate in conflict detection, making them difficult to use [24]. Probabilistic techniques seek to strike a balance between the simplistic nominal approach and the conservative worst-case approach by modeling uncertainties so that the probabilities of possible future trajectories can be calculated and ranked [24]. The probabilistic approach is thus the most general one: using a probabilistic model, the nominal approach corresponds to the case in which the aircraft is predicted to follow a (maximum likelihood) trajectory with probability of one, while the worse-case approach corresponds to the case in which the aircraft will follow a set of predicted trajectories with equal likelihood. Recent literature has been focused on probabilistic techniques. Particularly, models that are learned from historical data – such as Markov Models (MMs) [4, 12, 25] and Recurrent Neural Networks (RNNs) [38] – have gained considerable interest because of their effectiveness in modeling sequences for language processing [22, 23]. Moreover, the rapid development of sensor systems (such as GNSSs) used to provide geo-location data is making it more feasible to build such models, as they often require large amounts of training data. For example, the recent deployment of space-based Automatic Dependent Surveillance-Broadcast (ADS-B) [29] extends the coverage of air traffic information to the remaining 70% of the world's airspace (remote, polar and oceanic regions) that traditional ground-based ADS-B receivers are unable to cover [1]. In this thesis, we focus on the use of Markov Models for Trajectory Prediction, and leave Recurrent Neural Networks for future exploration and comparison after we are able to gather more training data.

Ayhan & Samet [4] proposed the use of Hidden Markov Models (HMMs) for Trajectory Prediction by considering the airspace as a 3D grid network, where

each point in the grid is the location of a weather observation (*i.e.* a vector of weather parameter values for temperature, wind speed, and wind direction). By building hypothetical cubes around these grid points, the airspace becomes a set of spatio-temporal data cubes. Each historical trajectory was aligned with the grid to be converted into a sequence of spatio-temporal data cubes, and each weather parameter value for each cube – temperature, wind speed, and wind direction – was assigned a bucket based on pre-defined intervals for each parameter, with 6, 5, and 8 possible buckets respectively. An HMM was built where states correspond to grid points, a transition probability represents the probability of an aircraft transitioning from one state to another along its aligned trajectory, an emission probability represents the probability of a set of weather parameters being observed at a particular state, and an initial probability represents the probability of an aligned trajectory beginning at a particular state. Experimental evaluation was performed by drawing 23 test trajectories with replacement from the 474 historical trajectories. Using the date of a test trajectory and the median flight duration of 78 minutes, each historical trajectory was treated as if it was flown on the same day as the test trajectory, to obtain a total of 474 sequences of 78 weather observations. Centroid weather observations for each of the 78 time instances were joined together [5] to form a single input observation sequence for the HMM, and were then passed through the Viterbi algorithm [22] to find the most likely sequence of hidden states, representing the sequence of spatio-temporal data cubes that form the prediction of the test trajectory.

As the approach proposed by Ayhan & Samet [4] was intended to be used for a *pre*-flight prediction of the complete aircraft trajectory from take-off to landing, it has the following weaknesses when applied for the online prediction of *mid*-flight aircraft trajectories:

- **Predicted trajectories always follow the median flight path**
  Based on the median flight duration of 78 minutes, the input to the HMM was a sequence of 78 weather observations, and the output was a sequence of 78 spatio-temporal data cubes representing the locations that the aircraft should be in for each minute of the flight. The flights for the same route

typically follow the same flight plan, detailing the most direct path from the departure airport to the arrival airport. Because of this, there is an over-representation of trajectories in the spatio-temporal data cubes that lie along this median flight path, resulting in large transition probabilities for state transitions along this path in the HMM. Also, the emission probabilities for each state are learned according to the historical distribution of weather parameter buckets in its corresponding spatio-temporal data cube. As the distributions of weather parameter buckets are specific to each data cube (especially for temperature parameter), the weather observations chosen for the input observation sequence are likely to have large emission probabilities for the states corresponding to the data cubes in which they are observed, and smaller or zero emission probabilities for other states. The proposed approach of joining the centroid weather observations of the 474 sequences of 78 weather observations will almost always generate the sequence of 78 weather observations that is observed along the median flight path, and because of the large transition and emission probabilities pertaining to this sequence of weather observations, the predicted trajectory will almost always be a sequence of 78 spatio-temporal data cubes along the same median flight path, which is not useful at all as it has already been described in the flight plan. Moreover, the assumption that the predicted trajectory is a sequence of 78 spatio-temporal data cubes contradicts the authors' claim that the proposed approach is able to providing accurate arrival metering: what it actually does is to make a prediction of the flight duration based on the median duration of all historical flights, without considering any environmental factors that could result in a delay of the flight.

- **Predicted trajectories do not improve offline flight planning**

  The approach was proposed with the aim of improving flight planning by calculating the optimal path for a flight trajectory, taking into account the weather parameters along possible flight trajectories for the current day of the flight. However, finding the optimal flight path of a single aircraft, without taking into consideration the other aircraft flying in the vicinity or pos-

sible congestion at the arrival airport, is akin to performing optimization without constraints: during its flight, the aircraft is bound to end up in conflict with another aircraft or face a congested airport at the destination, and be subjected to adjustment or re-routing by ATC. This is why the flight plan only provides a brief description of the flight path for the aircraft, and leaves it to the pilot to follow the instructions issued by ATC during the actual flight. Moreover, the output is a spatial trajectory that does not provide any suggestion of air-speed or heading: it does not provide instructions to the pilot on what speed or flight level to fly the aircraft at, and when or where to turn.

- **Predicted trajectories cannot be used for online monitoring**

  The proposed approach predicts the trajectory of an aircraft only once, before it takes off from the departure airport. This is different from online trajectory prediction, where predictions are made at regular intervals for as long as the aircraft is in the air. Even though the authors claimed that "the proposed solution can be adapted for both tactical and strategic trajectory prediction" [4], the adaptation to an online solution (*i.e.* tactical/online trajectory prediction) is not straight-forward. For adaptation, the HMM could be extended such that a set of initial probabilities is learned for each minute into the flight, then, the trajectory of an aircraft that has been flying for *e.g.* 15 minutes can be predicted by starting at the initial probabilities for the 15^{th} minute. However, without incorporating the current position of the aircraft into the modeling of the HMM, the predicted trajectory may not necessarily begin from the current position of the aircraft.

Chapter 3

Basic Definitions

This chapter defines the basic concepts that are used for the rest of this paper, and introduces the problem definition. Because of the shared application domain, the definitions contain some similarities with those in Ayhan & Samet [4].

3.1 Geospatial Concepts

The World Geodetic System 84 (WGS84) defines a standard coordinate system for the Earth, serving as a reference for the Global Positioning System (GPS), a global navigation satellite system that provides accurate positional and time information to people or objects equipped with GPS receivers anywhere near the surface of the Earth.

Definition 3.1. Based on the WGS84 standard, a **geographical position** *gp* near the surface of the Earth can be represented using 3 coordinates,

$$gp = (latitude, longitude, altitude).$$

Definition 3.2. A **positional update** *pos*, provided by a GPS receiver, represents the geographical position *gp* of a particular aircraft at a particular timestamp *ts*,

$$pos = (gp, ts).$$

Definition 3.3. The *continuous* sequence of positions that form the exact path of travel of an aircraft can be called an original trajectory. As the original trajectory can be infinitely long and immeasurable, we use an approximate representation

of it, called a **raw trajectory**. A raw trajectory Trj_{ac} is a finite sequence of positional updates of a particular aircraft *ac*, captured over a sequence of timestamps in increasing order,

$$Trj_{ac} = (pos_1, \ldots, pos_n), n \in \mathbb{Z}^+, n \geq 2,$$

where each position is

$$pos_i = (gp_i, ts_i), \forall i \in [1, n].$$

3.2 Reference Grid Concepts

There is an infinite number of possible representations of geographical positions over the continuous geographical coordinates latitude, longitude and altitude. In order to simplify the space of possible representations to a discrete and finite set, we perform a mapping of geographical positions to standard reference points in a reference grid. Further to this, to enable the use of weather data in our work, we use the 2-dimensional reference grid of the Rapid Refresh (RAP) weather forecasting system by the National Oceanic and Atmospheric Administration (NOAA) [7]. Under this mapping, geographical positions over the whole continental United States (CONUS) are first projected to local coordinates using the Lambert Conformal Conic Projection [40], before being mapped to a 2-dimensional reference point in the planar reference grid. For the vertical dimension, RAP uses 50 vertical levels following a sigma coordinate system [7], which is problematic for our purposes as the vertical levels are not uniformly separated and the values for all 50 levels vary depending on the current surface pressure at the reference point in the grid. Therefore, we use the standard in aviation [9] to divide the vertical dimension into uniformly spaced intervals based on pressure altitude[1]. Weather parameters for each altitude interval follow that of the RAP vertical level closest to the center of the interval.

[1]Pressure altitude, though expressed in feet, is a measurement of atmospheric pressure at a certain height, and conversion of units between millibars and feet can be done using a formula published by NOAA [27].

Our 3-dimensional reference grid is consequently a 3-dimensional array of geographical positions,

$$rp = (rp\_latitude, rp\_longitude, rp\_altitude),$$

spaced 13.545 km apart in a 451 $\times$ 337 configuration in the horizontal plane and spaced 2000 ft apart over 22 levels in the vertical dimension. The geographical volume of interest has thus been divided into discrete, evenly spaced volumes. Each volume, called a grid cube *gc*, is represented by a corresponding reference point *rp* at its center.

Given the reference grid, a raw trajectory *Trj* can be transformed into an aligned trajectory $\overline{Trj}$, by replacing the geographical position of each positional update in the raw trajectory with the reference point closest to it.

In what follows, we refer to the reference point *rp* representing a grid cube *gc* as $rp = ref(gc)$. We also refer to the set of all grid cubes as G.

3.3 Weather Concepts

RAP is an hourly updated weather forecasting system that provides a location-specific forecast of hourly weather conditions up to the next 19 hours[2] [7]. Due to the accumulation of errors in numerically computing the interactions of environmental variables over time, the accuracy of weather forecasting decreases as the forecast hour into the future increases [43]. As the aim of our work is to examine the effect of current weather conditions on the trajectory of an aircraft, we will be using only the forecast of weather conditions for the current hour.

Definition 3.4. A **weather vector** $wv_{dy,hr,gc}$, describing the weather condition in a particular grid cube *gc* at a particular hour *hr* of a day[3] *dy*, consists of 4 weather parameters, namely specific humidity $sh_{dy,hr,gc}$, temperature (Kelvins) $tk_{dy,hr,gc}$, wind

[2]For the current hour and 18 hours into the future.
[3]We use the terms **day** and **date** interchangeably.

speed $ws_{dy,hr,gc}$ and wind direction $wd_{dy,hr,gc}$,

$$wv_{dy,hr,gc} = \begin{bmatrix} sh_{dy,hr,gc} \\ tk_{dy,hr,gc} \\ ws_{dy,hr,gc} \\ wd_{dy,hr,gc} \end{bmatrix}.$$

Given a timestamp *ts*, we also write $wv_{ts,gc}$ as a shortcut for $wv_{date(ts),hour(ts),gc}$.

3.4 Weather Binning Concepts

We adopt a binning approach for discretizing weather vectors, such that each continuous weather vector *wv* can be mapped to a corresponding vector of discrete bin values *wb* to be represented in a Markov Model. Because of the varying distributions of the weather parameters over different grid cubes, each grid cube requires a different set of bin intervals for each of the weather parameters. The wind direction parameter has a circular distribution, and is best divided using the 4 cardinal directions (north, south, east, west), giving a total of 4 bin intervals of 90 degrees each, as shown in Table 3.1. As for the parameters specific humidity, temperature, and wind speed, we bin them such that values representing extreme weather conditions fall into different bins from values representing common weather conditions. We first obtain mean and standard deviation values for each parameter in each grid cube $gc \in G$, which we then use to determine 3 bin intervals by splitting the range of values at the first standard deviation from their means, as shown in Table 3.2.

In what follows, we refer to the bin vector *wb* mapped from a weather vector *wv* as $wb = bin(wv)$. We also refer to the set of all possible bin vectors as W.

Table 3.1: Bin intervals for wind direction

Bin intervals for wind direction (*wd*), in degrees	
Bin no.	Interval
1	$315(=-45) < wd \leq 45$
2	$45 < wd \leq 135$
3	$135 < wd \leq 225$
4	$225 < wd \leq 315(=-45)$

Table 3.2: Bin intervals for specific humidity, temperature and wind speed

Bin intervals for specific humidity (sh) in each grid cube $gc \in G$, same calculations apply for temperature (tk) and wind speed (ws)	
Bin no.	Interval
1	$\left(\mu_{sh,gc} - \sigma_{sh,gc}\right) < sh < \left(\mu_{sh,gc} + \sigma_{sh,gc}\right)$
2	$sh \geq \left(\mu_{sh,gc} + \sigma_{sh,gc}\right)$
3	$sh \leq \left(\mu_{sh,gc} - \sigma_{sh,gc}\right)$

3.5 Problem Definition

The Online Trajectory Prediction (TP) Problem. Given a set of D historical trajectories $\{Trj_1, ..., Trj_D\}$ of flights between two airports, as well as the trajectory of an aircraft currently on its journey since its departure timestamp ts_{dep} until the current timestamp ts_{now}, *i.e.* $(pos_{ts_{dep}}, ..., pos_{ts_{now}})$, **our goal is to predict the most probable trajectory of the aircraft for the next L minutes**, *i.e.* $(\widehat{pos}_{ts_{now+1}}, ..., \widehat{pos}_{ts_{now+L}})$. The time L is called the look-ahead time[4]. Also, we note that this is for *mid*-flight, *online* predictions, which are bound to be more useful in ATC applications than *pre*-flight predictions.

[4]Most conflict detection applications are interested in a look-ahead time of ten to twenty minutes [15].

Chapter 4

Prediction Models

This chapter provides the theoretical foundation of the prediction models. It gives an overview of Hidden Markov Models (HMMs), followed by the mapping of the TP problem to an HMM. It then presents the formulae for the estimation of probabilities in the HMM, followed by four proposed approaches for HMM decoding to obtain the predicted trajectory. It concludes with an illustration of how the four proposed approaches work, using a single example.

4.1 Hidden Markov Models (HMMs)

An HMM is specified by the following components:

- A set of N **possible states**, $S = \{s_1, \ldots, s_N\}$.
- A **transition probability matrix**,

$$A = \begin{bmatrix} a(s_1, s_1) & \ldots & a(s_1, s_N) \\ \vdots & \ddots & \vdots \\ a(s_N, s_1) & \ldots & a(s_N, s_N) \end{bmatrix},$$

  where each value $a(s_i, s_j)$ represents the probability of state s_i transitioning to state s_j, such that $\sum_{s_j \in S} a(s_i, s_j) = 1, \forall s_i \in S$.
- A set of M **possible observations**, $U = \{u_1, \ldots, u_M\}$.
- An **emission probability matrix**,

$$B = \begin{bmatrix} b(s_1, u_1) & \ldots & b(s_1, u_M) \\ \vdots & \ddots & \vdots \\ b(s_N, u_1) & \ldots & b(s_N, u_M) \end{bmatrix},$$

where each value $b(s_j, u_k)$ represents the probability of state s_j producing observation u_k, such that $\sum_{u_k \in U} b(s_j, u_k) = 1, \forall s_j \in S$.

- Special **start and end states**, s_0 and s_E, respectively, that are not associated with observations, with transition probabilities out of the start state $a(s_0, s_1), \ldots, a(s_0, s_N)$, as well as transition probabilities into the end state $a(s_1, s_E), \ldots, a(s_N, s_E)$.

HMMs have to address three fundamental problems: Likelihood Computation, Decoding, and Learning [22]. For these problems, the input to the model is an **observation sequence** $O = (o_1, \ldots, o_T)$, *i.e.* a sample sequence of T observations, such that $o_t \in U, \forall t \in [1, T]$.

In the HMM Decoding Problem, the Viterbi algorithm [22] is commonly used to find the **most probable hidden state sequence** $\widehat{Q} = (\widehat{q}_1, \ldots, \widehat{q}_T)$ for a given input observation sequence, *i.e.* the most probable sequence of T hidden states based on an input observation sequence O, such that $\widehat{q}_t \in S, \forall t \in [1, T]$.

4.2 Mapping the TP Problem to an HMM

We can model a particular flight route by probabilities of state changes in a Markov Model, based on historical trajectories of the flight route across the geographical space. We can then predict the most probable trajectory of the flight by finding the most probable state sequence, using the Markov Model, through the following mapping:

- The set of **possible states** S corresponds to the set of grid cubes G, each of which is represented by a reference point. Since there are $451 \times 337 \times 22 = 3,343,714$ reference points, we have a total of $N = 3,343,714$ possible states. A state $s_i \in S$ is said to be true for a particular aircraft at a particular time, when the aircraft is in the corresponding grid cube $gc_i \in G$ at that time.

- The **transition probability** $a_t(s_i, s_j)$ represents the probability of the aircraft moving from grid cube gc_i, corresponding to s_i, to grid cube gc_j, corresponding to s_j, after t minutes.

We have extended our model from a simple Markov Model by adding the subscript t in the transition probability, which we call a **timestep**. For example, $t = 3$ means a timestep of 3 minutes. This is intended as a work-around for an inherent limitation of HMMs in that there is no explicit representation of the time duration in each state [20], to allow us to utilize the temporal dimension in trajectory data for building our prediction models.

Also, by extending to a Hidden Markov Model consisting of states which are partially observable through observations, we can incorporate the influence of weather conditions on the trajectory of the aircraft by treating weather vectors as observations produced as the aircraft transitions through the hidden states:

- The set of **possible observations** U corresponds to the set of bin vectors W, each representing an interval of possible weather vectors. Since there are $4 \times 3 \times 3 \times 3 = 108$ bins, we have a total of $M = 108$ possible observations. An observation $u_k \in U$ is said to be true for a particular aircraft in a particular state s_j at a particular time, when the aircraft is in the corresponding grid cube gc_j at that time, and is experiencing the weather condition represented by wv that falls into the corresponding bin vector $bin\,(wv) = wb_k \in W$.
- The **emission probability** $b(s_j, u_k)$ represents the probability of an aircraft experiencing a weather condition that falls into bin vector wb_k, corresponding to u_k, when the aircraft is in grid cube gc_j, corresponding to s_j.

We also need to determine the start and end states in the HMM for the TP problem. Based on the problem definition, predictions shall be made anytime during the flight of the aircraft, with the implication that the start state should represent the current location of the aircraft. Because of this, the start state in our model is not a special state, but rather, a state belonging to the set of possible states, $s_i \in S, i \in [1, N]$. We write $q_{ts_{now}} = s_i$, to indicate and "select" the start state as the state at the current timestamp ts_{now}. For a given timestep t, transition probabilities out of the start state follow those for state s_i, *i.e.* $a_t(s_i, s_1), \ldots, a_t(s_i, s_N)$. The end state in our model is a special state s_E that is not in the set of possible states S. Since there is no reason to prefer any particular state to be the last state in the predicted hidden state sequence (*i.e.* the state before the special end state s_E), we

omit transition probabilities into the end state, and also the end state itself, from our model.

In summary, given look-ahead time L, and assuming we have learned (see the following section) the emission probability matrix and the transition probability matrices for each minute t into the future up to the look-ahead time (*i.e.* $\forall t \in [1, L]$), we have the following input and output to our model:

- Input: start state $q_{ts_{now}}$ and observation matrix O of size $N \times L$. Instead of taking an observation sequence, our model takes as input the start state $q_{ts_{now}} \in S$ at the current timestamp ts_{now}, its corresponding grid cube $gc_{ts_{now}}$, as well as the weather conditions observed in all N grid cubes for the following L minutes, $O = [o_{j,t}], j \in [1, N], t \in [1, L]$, such that each observation $o_{j,t} = u_k \in U$ corresponds to a bin vector $wb_k = bin(wv_{ts_{now}+t, gc_j}) \in W, gc_j \in G, t \in [1, L]$, where we assume timestamps to be whole minutes.
- Output: most probable hidden states $\widehat{q}_{ts_{now}+1}, \ldots, \widehat{q}_{ts_{now}+L}$. We find the most probable state $\widehat{q}_{ts_{now}+t}$ for each minute t into the future up to the look-ahead time of $L = 20$ minutes, $t \in [1, L]$, and concatenate them in ascending temporal order to form the sequence of most probable hidden states, $\widehat{Q} = (\widehat{q}_{ts_{now}+1}, \ldots, \widehat{q}_{ts_{now}+L})$, which can be mapped into a predicted trajectory for the next L minutes, by assigning as positions the reference points of grid cubes corresponding to the predicted states, *i.e.* $\widehat{Trj} = (\widehat{pos}_{ts_{now}+1}, \ldots, \widehat{pos}_{ts_{now}+L}) = (ref(\widehat{gc}_{ts_{now}+1}), \ldots, ref(\widehat{gc}_{ts_{now}+L}))$.

4.3 Estimating Probabilities for the HMM

Our estimation of transition probabilities follows the usual approach of counting occurrences, using a training dataset consisting of D aligned historical trajectories $\{\overline{Trj}_1, \ldots, \overline{Trj}_D\}$.

For a timestep of t minutes, $t \in [1, L]$, $a_t(s_i, s_j)$ represents the probability of state $s_i \in S$ transitioning to state $s_j \in S$ after t minutes, and can be estimated by counting, how many times aircrafts on historical flights were in grid cube gc_j, corresponding to state s_j, when they were in grid cube gc_i, corresponding to state

s_j, t minutes earlier, *i.e.*,

$$\begin{aligned} a_t(s_i, s_j) &= P((\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}} \mid (\mathit{ref}(gc_i), \mathit{ts} - t) \in \overline{\mathit{Trj}}\ ) \\ &= \frac{|\ \{\overline{\mathit{Trj}} \mid (\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}} \wedge (\mathit{ref}(gc_i), \mathit{ts} - t) \in \overline{\mathit{Trj}}\ \}\ |}{|\ \{\overline{\mathit{Trj}} \mid (\mathit{ref}(gc_i), \mathit{ts} - t) \in \overline{\mathit{Trj}}\ \}\ |}, \end{aligned} \tag{4.1}$$

where $(\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}}$ denotes that an aligned trajectory $\overline{\mathit{Trj}}$ contains a position $\mathit{ref}(gc_j)$ (*i.e.* the reference point of grid cube gc_j, corresponding to state s_j) with timestamp *ts*.

As for the estimation of emission probabilities, we follow a similar approach. However, given a total of $M = 108$ bins where each weather vector can fall into, a huge amount of data would be required to estimate the emission probabilities without obtaining zero probability values. To mitigate this situation, we perform Laplace Smoothing in our estimation by adding a pseudo-count of 1 for each bin (*i.e.* **Add-One Smoothing**) [26].

$b(s_j, u_k)$ represents the probability of state $s_j \in S$ producing observation $u_k \in U$, and can be estimated by counting (with Add-One Smoothing), how many times aircrafts on historical flights experienced a weather condition that falls into bin vector wb_k, corresponding to observation u_k, when they were in grid cube gc_j, corresponding to state s_j, *i.e.*,

$$\begin{aligned} b(s_j, u_k) &= P(\mathit{bin}\,(\mathit{wv}_{\mathit{ts},gc_j}) = \mathit{wb}_k \mid (\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}}\ ) \\ &\approx \frac{|\ \{\overline{\mathit{Trj}} \mid \mathit{bin}\,(\mathit{wv}_{\mathit{ts},gc_j}) = \mathit{wb}_k \wedge (\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}}\ \}\ | + 1}{|\ \{\overline{\mathit{Trj}} \mid (\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}}\ \}\ | + 108}, \end{aligned} \tag{4.2}$$

where $(\mathit{ref}(gc_j), \mathit{ts}) \in \overline{\mathit{Trj}}$ denotes that an aligned trajectory $\overline{\mathit{Trj}}$ contains a position $\mathit{ref}(gc_j)$ (*i.e.* the reference point of grid cube gc_j, corresponding to state s_j) with timestamp *ts*, and $\mathit{bin}\,(\mathit{wv}_{\mathit{ts},gc_j}) = \mathit{wb}_k$ denotes that the weather condition $\mathit{wv}_{\mathit{ts},gc_j}$ in grid cube gc_j at timestamp *ts* falls into the bin vector wb_k.

4.4 Decoding the HMM: Four Approaches

We propose four approaches for decoding the HMM we have built for Trajectory Prediction, based on utilizing the multi-timestep transition probabilities learned from historical trajectories. Essentially, when propagating probabilities from states

at timestamp $ts_{now} + t - 1$ to states at timestamp $ts_{now} + t$, where $t \in [1, L]$, the four approaches use different combinations of two transition probabilities:

- 1-minute transition probability from a state s_i at timestamp $ts_{now} + t - 1$ to a state s_j at timestamp $ts_{now} + t$, *i.e.* $a_1(s_i, s_j)$.
- t-minute transition probability from the start state $q_{ts_{now}}$ at timestamp ts_{now} to a state s_j at timestamp $ts_{now} + t$, *i.e.* $a_t(q_{ts_{now}}, s_j)$.

4.4.1 The Viterbi Approach

The first approach applies the Viterbi algorithm [21], using dynamic programming based on **only 1-minute transitions** to find the most likely hidden state sequence. This is done through finding the Viterbi probabilities for all possible states, recursively in increasing order of minutes from the current timestamp.

The **Viterbi probability** $vt_t(s_j)$ for a state s_j at timestamp $ts_{now} + t$, $t \in [1, L]$, can be computed by finding the maximum, among all states $s_i \in S$, the product of the Viterbi probability $vt_{t-1}(s_i)$ of state s_i at timestamp $ts_{now} + t - 1$, the 1-minute transition probability $a_1(s_i, s_j)$ from state s_i to state s_j, and the emission probability $b(s_j, o_{j,t})$ of observation $o_{j,t}$ in state s_j at timestamp $ts_{now} + t$, *i.e.*,

$$vt_t(s_j) = b(s_j, o_{j,t}) \times \max_{s_i \in S} \big(vt_{t-1}(s_i) \times a_1(s_i, s_j)\big). \tag{4.3}$$

The Viterbi algorithm uses a **backtrace pointer** $bt_t(s_j)$ for each state s_j at each timestamp $ts_{now} + t$, $t \in [2, L]$, to keep track of the best state s_i in the previous timestamp $ts_{now} + t - 1$ that led to the current state s_j, and by following the backtrace pointer all the way to the beginning, and then concatenating the states along this path, the most likely hidden state sequence can be obtained.

Based on the description of our HMM presented so far, we give a formal definition of the Viterbi recursion as follows:

Initialization:

$$\begin{aligned} vt_1(s_j) &= b(s_j, o_{j,t}) \times a_1(q_{ts_{now}}, s_j), && \forall s_j \in S, \\ bt_1(s_j) &= \emptyset, && \forall s_j \in S, \end{aligned}$$

Recursion:

$$vt_t(s_j) = b(s_j, o_{j,t}) \times \max_{s_i \in S} \big(vt_{t-1}(s_i) \times a_1(s_i, s_j)\big), \qquad \forall s_j \in S, \forall t \in [2, L],$$
$$bt_t(s_j) = b(s_j, o_{j,t}) \times \operatorname*{argmax}_{s_i \in S} \big(vt_{t-1}(s_i) \times a_1(s_i, s_j)\big), \quad \forall s_j \in S, \forall t \in [2, L],$$

Termination:

$$\textbf{best probability: } \max_{s_i \in S} \big(vt_L(s_i)\big),$$
$$\textbf{start of backtrace: } \operatorname*{argmax}_{s_i \in S} \big(bt_L(s_i)\big).$$

4.4.2 The Multi-Timestep Approach

The second approach, as presented in Pan, Nascimento, & Sander [34], uses **only t-minute transitions** to find the most likely hidden state at each minute given the start state, and thus does not require dynamic programming like the Viterbi approach.

For this approach, the most probable state $\widehat{q}_{ts_{now}+t}$ at timestamp $ts_{now} + t$ is the state $s_j \in S$ that maximizes the product of the t-minute transition probability $a_t(q_{ts_{now}}, s_j)$ from the start state $q_{ts_{now}}$ to state s_j, and the emission probability $b(s_j, o_{j,t})$ of observation $o_{j,t}$ in state s_j at timestamp $ts_{now} + t$, *i.e.*,

$$\widehat{q}_{ts_{now}+t} = \operatorname*{argmax}_{s_j \in S} \big(b(s_j, o_{j,t}) \times a_t(q_{ts_{now}}, s_j)\big). \tag{4.4}$$

As compared to the $O(N^2 L)$ complexity of the Viterbi algorithm, the advantage of the Multi-Timestep approach is that it has a complexity of $O(NL)$.

4.4.3 The Constrained Viterbi Approach

The third approach is based on applying the Viterbi algorithm as described previously. **It uses 1-minute transition probabilities** in calculating the Viterbi probabilities, and **also uses t-minute transition probabilities as a constraint** to limit the number of possible states being considered for each minute.

This is achieved by adding the term $sgn(a_t(q_{ts_{now}}, s_j))$ to the Viterbi probability in Equation 4.3. This term uses the **sign function** to set the Viterbi probability of the state s_j at timestamp $ts_{now} + t$ to zero when the t-minute transition probability $a_t(q_{ts_{now}}, s_j)$ from the start state is zero, and to multiply the Viterbi probability by 1

(essentially doing nothing to it) when the t-minute transition probability is more than zero, *i.e.*,

$$vt_t(s_j) = b(s_j, o_{j,t}) \times sgn\big(a_t(q_{ts_{now}}, s_j)\big) \times \max_{s_i \in S} \big(vt_{t-1}(s_i) \times a_1(s_i, s_j)\big). \quad (4.5)$$

The same modification is made to the Viterbi backtrace, while the Initialization and Termination clauses in the Viterbi recursion remain unchanged.

4.4.4 The Constrained Multi-Timestep Approach

The fourth approach, inspired by the Multi-Timestep approach, is also based on applying the Viterbi algorithm, but switches the roles of the two transition probabilities. **It uses t-minute transition probabilities** in calculating the Viterbi probabilities, and **also uses 1-minute transition probabilities as a constraint** to ensure that there is a path through the hidden state sequence found by the algorithm.

The **sign function** $sgn\big(a_1(s_i, s_j)\big)$ is now applied to the 1-minute transition probability from Equation 4.3, while the t-minute transition probability $a_t(q_{ts_{now}}, s_j)$ is now used to calculate the magnitude of the Viterbi probability. With this modification, the equation follows closely that of Equation 4.4 for the Multi-Timestep approach, but in calculating the Viterbi probability of the state s_j at timestamp $ts_{now} + t$, only considers states $s_i \in S$ in the previous timestamp $ts_{now} + t - 1$ that have a positive 1-minute transition probability $a_1(s_i, s_j)$ to state s_j, *i.e.*,

$$vt_t(s_j) = b(s_j, o_{j,t}) \times a_t(q_{ts_{now}}, s_j) \times \max_{s_i \in S} \big(vt_{t-1}(s_i) \times sgn\big(a_1(s_i, s_j)\big)\big). \quad (4.6)$$

The same modification is made to the Viterbi backtrace, while the Initialization and Termination clauses in the Viterbi recursion remain unchanged.

4.5 Working through an Example

We illustrate how the four proposed approaches work, using a single example comprising 11 states over 3 minutes as shown in Figure 4.1, where 1-minute, 2-minute, and 3-minute transition probabilities are represented by red dashed lines, blue dotted lines, and green solid lines respectively. Emission probabilities have

been omitted from this example for the sake of simplicity. The prediction starts from $s_1 = q_{ts_{now}}$, with the following transition probabilities:

$$
\begin{aligned}
&a_1(s_1, s_2) = 0.3, \quad a_1(s_1, s_3) = 0.3, \quad a_1(s_1, s_4) = 0.4,\\
&a_1(s_2, s_5) = 1.0,\\
&a_1(s_3, s_5) = 0.33, \quad a_1(s_3, s_6) = 0.66,\\
&a_1(s_4, s_6) = 0.5, \quad a_1(s_4, s_7) = 0.5,\\
&a_1(s_5, s_8) = 0.25, \quad a_1(s_5, s_{11}) = 0.75,\\
&a_1(s_6, s_9) = 0.5, \quad a_1(s_6, s_{10}) = 0.5,\\
&a_1(s_7, s_{10}) = 1.0,\\
&a_2(s_1, s_5) = 0.4, \quad a_2(s_1, s_6) = 0.4, \quad a_2(s_1, s_7) = 0.2,\\
&a_3(s_1, s_8) = 0.14, \quad a_3(s_1, s_9) = 0.29, \quad a_3(s_1, s_{10}) = 0.57.
\end{aligned}
$$

Note that there is a state s_{11} that does not have a transition probability from the start state within 3 minutes.

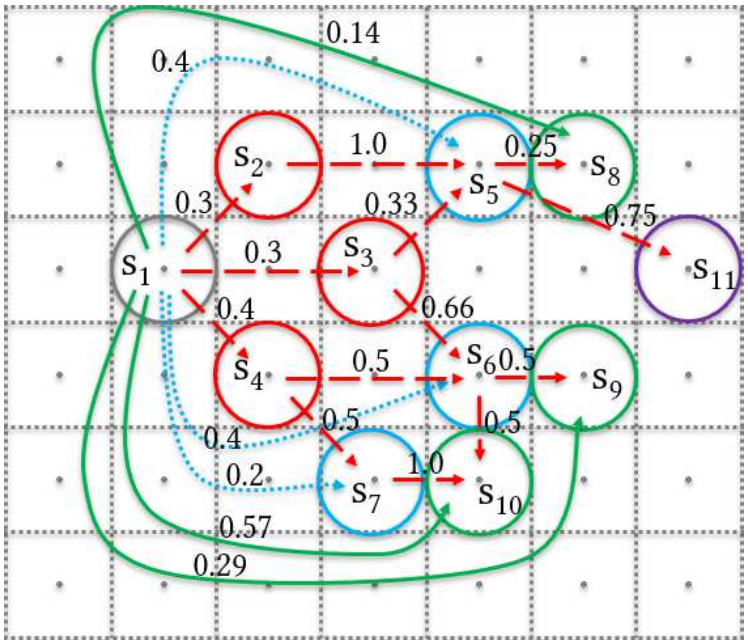

Figure 4.1: An example with 11 states and their transition probabilities

4.5.1 Using the Viterbi Approach

Based on the Viterbi approach, the following Viterbi probabilities are calculated:

$$
\begin{aligned}
vt_1(s_2) &= a_1(s_1, s_2) &&= 0.3,\\
vt_1(s_3) &= a_1(s_1, s_3) &&= 0.3,\\
vt_1(s_4) &= a_1(s_1, s_4) &&= 0.4,\\
vt_2(s_5) &= \max(vt_1(s_2) \times a_1(s_2, s_5), vt_1(s_3) \times a_1(s_3, s_5)) &&= 0.3,\\
vt_2(s_6) &= \max(vt_1(s_3) \times a_1(s_3, s_6), vt_1(s_4) \times a_1(s_4, s_6)) &&= 0.2,\\
vt_2(s_7) &= vt_1(s_4) \times a_1(s_4, s_7) &&= 0.2,\\
vt_3(s_8) &= vt_2(s_5) \times a_1(s_5, s_8) &&= 0.075,\\
vt_3(s_9) &= vt_2(s_6) \times a_1(s_6, s_9) &&= 0.1,\\
vt_3(s_{10}) &= \max(vt_2(s_6) \times a_1(s_6, s_{10}), vt_2(s_7) \times a_1(s_7, s_{10})) &&= 0.2,\\
vt_3(s_{11}) &= vt_2(s_5) \times a_1(s_5, s_{11}) &&= 0.225.
\end{aligned}
$$

The most probable hidden state sequence found using the backtrace is s_2, s_5, s_{11}.

4.5.2 Using the Multi-Timestep Approach

The Multi-Timestep approach can sometimes result in state sequences that do not make sense when mapped to predicted trajectories. For example, if the transition probabilities are processed in the order they were presented above, the output sequence would be s_4, s_5, s_{10}. The trajectory represented by this sequence consists of very sharp turns, which is often unrealistic for civilian aircraft. The output sequence s_4, s_6, s_{10} is also possible, if the transition probabilities are processed in a different order.

4.5.3 Using the Constrained Viterbi Approach

The Constrained Viterbi approach will have the same Viterbi probabilities as the Viterbi approach above, except that since state s_{11} has a zero 3-minute transition probability from the start state, *i.e.* $a_3(s_1, s_{11}) = 0$, the Viterbi probability at $ts_{now}+3$ for state s_{11} is now zero, *i.e.* $vt_3(s_{11}) = 0$. The most probable hidden state sequence found using the backtrace will be s_4, s_7, s_{10} instead.

4.5.4 Using the Constrained Multi-Timestep Approach

The Contrained Multi-Timestep approach is an improvement from the Multi-Timestep approach in that it ensures that consecutive states in the output state sequence are joined together by non-zero 1-minute transitions. Based on the Contrained Multi-Timestep approach, the following Viterbi probabilities are calculated:

$$\begin{aligned}
vt_1(s_2) &= a_1(s_1, s_2) && = 0.3, \\
vt_1(s_3) &= a_1(s_1, s_3) && = 0.3, \\
vt_1(s_4) &= a_1(s_1, s_4) && = 0.4, \\
vt_2(s_5) &= a_2(s_1, s_5) \times \max(vt_1(s_2), vt_1(s_3)) && = 0.12, \\
vt_2(s_6) &= a_2(s_1, s_6) \times \max(vt_1(s_3), vt_1(s_4)) && = 0.16, \\
vt_2(s_7) &= a_2(s_1, s_7) \times vt_1(s_4) && = 0.08, \\
vt_3(s_8) &= a_3(s_1, s_8) \times vt_2(s_5) && = 0.0168 \\
vt_3(s_9) &= a_3(s_1, s_9) \times vt_2(s_6) && = 0.0464 \\
vt_3(s_{10}) &= a_3(s_1, s_{10}) \times \max(vt_2(s_6), vt_2(s_7)) && = 0.0912
\end{aligned}$$

The most probable hidden state sequence found using the backtrace is s_4, s_6, s_{10}.

4.5.5 The Intuition Behind the Four Approaches

From the example, it can be seen that the Viterbi approach behaves like a greedy search that aims to maximize the product of the 1-minute transition probabilities along the output state sequence. The weakness of this approach is that it ignores the start state, *i.e.* the current position of the aircraft, after the states for the first minute into the future have been considered. The Multi-Timestep approach, on the other hand, selects the most probable state at t minutes into the future by finding the state that has the greatest t-minute transition probability from the start state. As it ignores the possibility that consecutive states along the output hidden state sequence might not be located geographically close to each other, it can result in predicted trajectories that do not make physical sense, such as having sharp turns or simply 'teleporting' from one position to the next.

The Constrained Viterbi approach is an improvement from the Viterbi approach in that when considering a state at t minutes into the future, it enforces

the constraint that this state must have a positive t-minute transition probability from the start state. This ensures that states that are not 'reachable' from the start state in t minutes will not be considered as the predicted state for that minute. Similarly, the Constrained Mulit-Timestep approach is an improvement from the Multi-Timestep approach in that when considering a state at t minutes into the future, it enforces the constraint that this state must have a positive 1-minute transition probability from a state in the previous minute (and uses the Viterbi probability of that state in the previous minute in calculating the Viterbi probability of this state). This ensures that consecutive states along the output state sequence are 'reachable' from the previous state in 1 minute.

We also illustrate how the four proposed approaches calculate the probability of a state sequence differently, using an example shown in Figure 4.2, where the state and observation symbols in the notations have been omitted for simplicity.

Probability of a state sequence calculated by each approach	
Viterbi	$b' \times b'' \times b''' \times {a_1}' \times {a_1}'' \times {a_1}'''$
Constrained Viterbi	$b' \times b'' \times b''' \times {a_1}' \times {a_1}'' \times {a_1}''' \times sgn({a_1}') \times sgn({a_2}') \times sgn({a_3}')$
Multi-Timestep	$b' \times b'' \times b''' \times {a_1}' \times {a_2}' \times {a_3}'$
Constrained Multi-Timestep	$b' \times b'' \times b''' \times {a_1}' \times {a_2}' \times {a_3}' \times sgn({a_1}') \times sgn({a_1}'') \times sgn({a_1}''')$

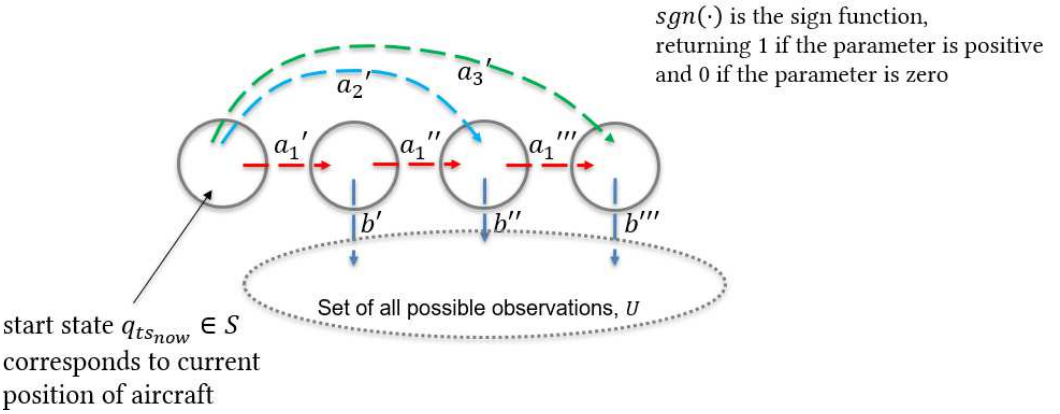

Figure 4.2: An example of the probability calculation for a state sequence

Chapter 5

Experimental Evaluation

This chapter describes the experimental evaluation of the prediction models. It gives an overview of the data and models used for comparison, followed by the accuracy metrics used and the process flow for building the models. It then presents an analysis of prediction time taken by the models, followed by an analysis of prediction accuracy. It concludes with two additional experiments for improving the models, which were conducted retrospectively.

5.1 Description of Data

The weather data used for our work comes from the Rapid Refresh (RAP) weather forecasting system developed by the National Oceanic and Atmospheric Administration (NOAA) [7, 36]. The website for the RAP system provides a link to an archive of data generated since May 01, 2012. A single file, recording the weather conditions over the whole of the continental United States, is archived for each hour of a day.

The trajectory data used for our work is based on the Automatic Dependent Surveillance Broadcast (ADS-B) technology, and comes from historical ADS-B data archived by the OpenSky Network [37, 41]. The data is stored in the form of positional updates, which are concatenated together based on flight number stored in the 'callsign' field. Trajectories for a particular flight route are found by checking if the start and end positions of a trajectory is near to the origin and destination

airports for the flight route[5].

We downloaded trajectory and weather data over a total of 25 months, from 2017-Jan to 2019-Jan (inclusive). The size of the trajectory data collected for each flight route, together with median flight times, are shown in Table 5.1 (LGA, ORD, and BOS are the IATA codes for LaGuardia Airport, Chicago O'Hare Airport, and Boston Logan Airport, respectively), while the break-down of the number of trajectories by month is shown in Figure 5.1. The small number (≤ 50) of trajectories in certain months, especially for the first few months of the dataset, can be attributed to poor sensor coverage, resulting in trajectories having large 'gaps' being dropped during data pre-processing. We present most of our comparisons based on the busiest flight route R1, as it has the largest number of trajectories.

Table 5.1: Meta-data of flight routes used in the experiments

Route code	Origin / Destination	Mean flight time (minutes)	Number of trajectories	Number of positions
R1	LGA-ORD	116	8036	917299
R2	ORD-LGA	97	4020	389777
R3	BOS-ORD	133	2876	379938
R4	ORD-BOS	108	2049	221820

Finally, all experiments were run on a computer with Intel Core i7-6700HQ 2.6GHz CPU and 8GB DDR4 Memory, using the Java software libraries GeoTools [14] for geospatial calculations and NetCDF [30] for reading RAP weather data.

5.2 Description of Models

We implemented two baseline models for comparison. The first, called Kinematics-Based Model (KBM), predicts the future trajectory of the aircraft based on its current track angle, ground speed and vertical rate. Using GeoTools [14], we make a prediction by projecting the current position of the aircraft L minutes into the future based on these kinematic parameters.

We also implemented a Median Trajectory Model (MTM) that was inspired by Ayhan & Samet [4], though our work is not directly comparable because of

[5]Flight numbers cannot serve as primary keys for flight routes, as airlines may re-use the same flight number for a different flight route.

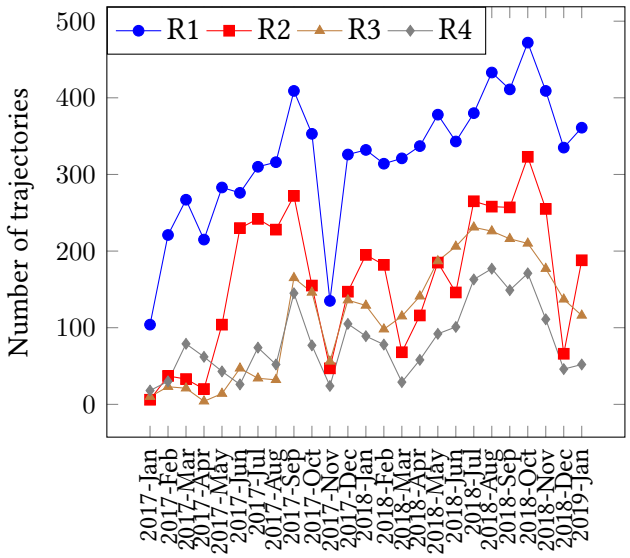

Figure 5.1: Number of trajectories for each month

the different intended application (*mid*-flight vs. *pre*-flight) and prediction time horizon (up to twenty minutes into the future *vs.* time since departure). For each trajectory in the training set of D historical trajectories, we first calculate the sum of its Dynamic Time Warping[6] (DTW) distances to all other trajectories. We then select the trajectory that has the smallest sum of DTW distances as the median trajectory. To make a prediction of the future trajectory of an aircraft based on its current position, we find the positional update on the median trajectory that is nearest to the current position of the aircraft. Based on a look-ahead time of L minutes, we then select the positional updates of the median trajectory that are up to L minutes after, and concatenate them to form the trajectory predicted by the MTM model.

In choosing the train-test split of the data, we tried two approaches. For the first approach, (*i.e.* **month-based training** approach) we attempt to capture possible seasonal differences in weather patterns. For each month in the second year (*e.g.* 2018-Jun), we train a prediction model using the same month in the previous year and its immediate neighbouring months (*e.g.* 2017-May, 2017-Jun, & 2017-Jul). Thus, we start our testing from the month of 2018-Feb (requiring 2017-Jan,

[6]Preliminary experiments have shown that Dynamic Time Warping provides a good distance measure between two trajectories for this use.

2017-Feb, & 2017-Mar for training) instead of 2018-Jan, and finish our testing on the final month of 2019-Jan (requiring 2017-Dec, 2018-Jan, & 2018-Feb for training) instead of 2018-Dec, spanning a total of 25 months of data. For the second approach, (*i.e.* **year-based training** approach) we train a prediction model using data for the first year (*i.e.* 2017-Feb to 2018-Jan), which we use for testing for each month in the second year (*i.e.* 2018-Feb, 2018-Mar, *etc.* until 2019-Jan).

Also, we seek to analyze the effect of incorporating weather information in our prediction models. We do this by using the HMMs we have presented, as well as their Observed Markov Model (OMM) equivalents, which are obtained by removing all emission probability terms $b(s_j, o_{j,t})$ from Equations 4.3, 4.4, 4.5, & 4.6. We present all the discussed models in Table 5.2.

Table 5.2: List of prediction models used in the experiments

Model code	Name of model	Training approach
KBM	Kinematics-Based Model	Training not required
MTM-1Y	Median Trajectory Model	Year-based
MTM-3M	Median Trajectory Model	Month-based
OMVB-1Y	Viterbi OMM	Year-based
OMVB-3M	Viterbi OMM	Month-based
OMVC-1Y	Contrained Viterbi OMM	Year-based
OMVC-3M	Contrained Viterbi OMM	Month-based
OMMB-1Y	Multi-Timestep OMM	Year-based
OMMB-3M	Multi-Timestep OMM	Month-based
OMMC-1Y	Contrained Multi-Timestep OMM	Year-based
OMMC-3M	Contrained Multi-Timestep OMM	Month-based
HMVB-1Y	Viterbi HMM	Year-based
HMVB-3M	Viterbi HMM	Month-based
HMVC-1Y	Contrained Viterbi HMM	Year-based
HMVC-3M	Contrained Viterbi HMM	Month-based
HMMB-1Y	Multi-Timestep HMM	Year-based
HMMB-3M	Multi-Timestep HMM	Month-based
HMMC-1Y	Contrained Multi-Timestep HMM	Year-based
HMMC-3M	Contrained Multi-Timestep HMM	Month-based

We note that our proposed approaches are unable to make predictions for some positions in the test set. This happens in cases when a test position falls into a grid cube that no position from a trajectory in the training set had fallen into. In order

to have a fair comparison with the baseline models, we cannot simply ignore these occurrences; we thus fall back on the KBM model to make predictions in these cases. We suggest an improvement to deal with this issue in the final section of this chapter.

5.3 Accuracy Metrics

We use the metrics presented by Paglione & Oaks [33] to evaluate the accuracy of our TP models by comparing predicted aircraft positions against actual aircraft positions.

We measure the error between a predicted position $\widehat{pos}_{ts_{now}+t}$ and the actual aircraft position $pos_{ts_{now}+t}$ for each minute t into the future, $t \in [1, L]$, up to the look-ahead time of $L = 20$ minutes. There are two types of errors that can be measured: horizontal error and vertical error. Horizontal error is the horizontal distance between the two positions $pos_{ts_{now}+t}$ and $\widehat{pos}_{ts_{now}+t}$, and is always positive. Vertical error is the difference in pressure altitude between the two positions $pos_{ts_{now}+t}$ and $\widehat{pos}_{ts_{now}+t}$, and is positive (negative) when the actual position $pos_{ts_{now}+t}$ is above (below) the predicted position $\widehat{pos}_{ts_{now}+t}$. We omit the sign when calculating statistics for vertical error, *i.e.* only the absolute value of the vertical error is considered.

We compute the mean and the standard deviation of the errors for all predictions within a test trajectory to obtain the intra-trajectory horizontal and vertical errors.

After that, we compute the mean and the standard deviation of all intra-trajectory horizontal and vertical errors within each test month (giving each trajectory an equal weight despite unequal number of predictions in each trajectory due to different flight times) to obtain the intra-month horizontal and vertical errors.

Finally, we compute the mean and the standard deviation of all intra-month horizontal and vertical errors (giving each month an equal weight despite unequal number of trajectories in each month) to obtain the overall mean and standard deviation of errors for the whole test set.

5.4 Process Flow

The processes for building our prediction models include data pre-processing, data discretization, and model training, as shown in Figure 5.2.

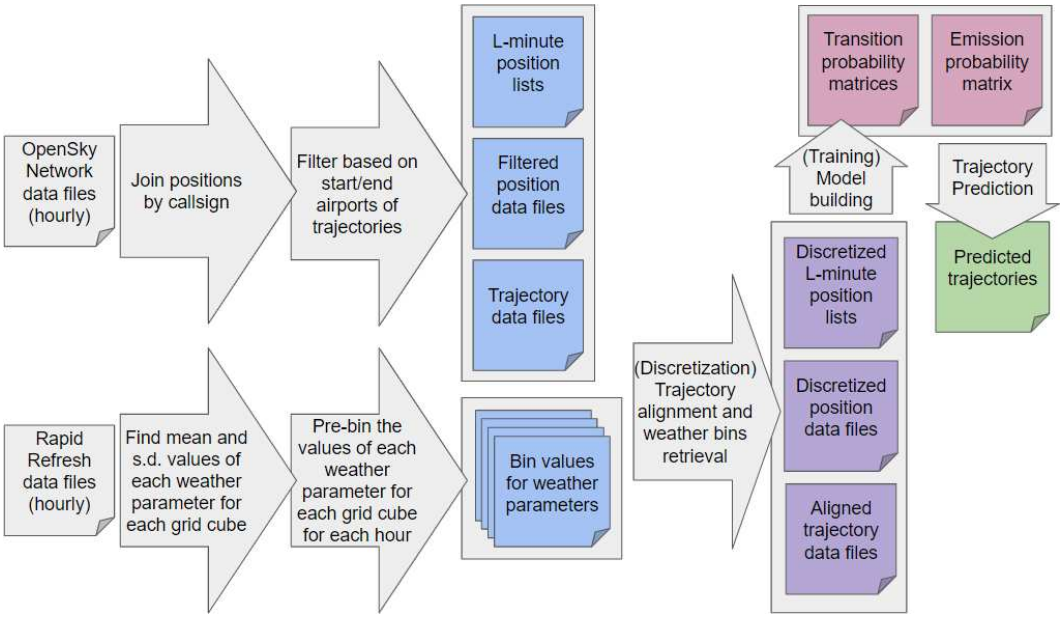

Figure 5.2: Overview of the process flow

The use of weather data by the HMM involves loading and computing a large amount of weather data computed hourly by the RAP system [7, 36] over the continental US for a period of 2 years. Through pre-binning of weather data, we can speed up the training and evaluation of models such that each weather vector is binned only once, and can be used by multiple models being evaluated on the same dataset. The size of the data that is loaded into memory during model training is also reduced, from about 26–30 MB (compressed) to 3 MB to store all bin vectors of a single hour. To do so, we first determine the mean and standard deviation values of each weather parameter for each grid cube over the whole training set. We then use these values to bin the weather vectors for each hour of the whole 2 years' weather dataset.

Aircraft movement data is collected and stored by the OpenSky Network [37, 41] in the form of positional updates, which we download in hourly portions for a period of 2 years. We join positional updates in chronological order to form tra-

jectories based on their callsign. As positions along a trajectory may be unevenly spaced due to poor sensor coverage, we discard trajectories that have 'gaps' in positional updates of longer than 5 minutes. Also, since positional updates along a trajectory can be within a few seconds apart from each other, we form a raw trajectory by keeping only positional updates that are spaced 1 minute apart. In cases where there is no positional update that is exactly 1 minute after a previous positional update, we select the one that is closest in either the positive and negative direction, *i.e.* in this order: 61 seconds, 59 seconds, 62 seconds, 58 seconds, *etc.* We also check the start and end positions of each raw trajectory, keeping only those that start and end at the origin and destination airports of the flight routes that we are interested in. Lastly, for each position on a raw trajectory, we select a list of t positions occuring $t = 1, \ldots, L$ minutes after, each allowing for a difference of no more than 15 seconds. This gives us a list of positions for each position along a raw trajectory to be used for counting the transition probability matrices during model training.

For data discretization, we map each positional update in a raw trajectory to the reference point nearest to it, giving us an aligned trajectory containing discretized positions (*i.e.* reference points with timestamps). For each discretized position, we also load its bin vector using its corresponding grid cube and timestamp, to be used for counting the emission probability matrix during model training.

Training the model involves counting of occurrences, using the list of positions and bin vector for each position along an aligned trajectory for estimating the transition probability matrices and emission probability matrix, according to Equations 4.1 & 4.2. Using the matrices, each position along a trajectory in the test set is treated as the current position, and used as input to the prediction model to find the most likely trajectory of the aircraft, based on Equation 4.3, 4.4, 4.5, or 4.6 (depending on which approach is being evaluated). Each predicted position along this predicted trajectory is compared to the actual position to measure the prediction error for each minute t into the future up to the look-ahead time L, $t \in [1, L]$, according to the accuracy metrics presented earlier.

5.5 Comparison of Prediction Time

The average time per prediction taken by each model is shown in Figure 5.3. The time for each model is calculated by taking the average of the running times for all test positions in the whole test set for all flight routes. Aided by the effectiveness of our data-preprocessing, all of the prediction models were able to make extremely fast predictions in the order of milliseconds, with the worst-performing model, the Viterbi-based HMM with 1-year training data (*i.e.* model HMVB-1Y) requiring an average of no more than 11.5 milliseconds per prediction.

The prediction times presented demonstrated large improvements from that presented in Pan, Nascimento, & Sander [34], where the model HMMB-3M took an average of 39.8 milliseconds compared to the improved 2.2 milliseconds. The Viterbi-based approaches were also infeasible, requiring prediction time in the order of minutes, compared to the improved 11.5 milliseconds by worst-performing model HMVB-1Y. There are two explanations to this.

In Pan, Nascimento, & Sander [34], the operations for reading and writing of test positions and predicted trajectories were included as part of the running time of a model. As all of the models presented [34] (*i.e.* KBM, MTM, and HMMB-3M) were able to make predictions efficiently, the average prediction time was inflated by the inclusion of running time for input and output. This was why all models presented [34] had average prediction times in the order of 30 to 50 milliseconds. For the average prediction times measured for this thesis, the running time for input and output has been excluded. This is not an improvement in prediction time, but rather a change in how prediction time is measured.

Secondly but more importantly, substantial improvements have been made through using **more efficient data structures**. With a total of $N = 3,343,714$ possible states, a transition probability matrix $a_t(s_i, s_j)$ for a particular timestep t would require a total of $N \times N \approx 1.1$ trillion elements. If each element is a floating point number taking up 4 bytes, it would require 4.4 TB of computer memory to store the matrix. In Pan, Nascimento, & Sander [34], this was handled by reducing the number of possible states N through pre-calculating the actual minimum

and maximum indices of grid cubes that are actually used in the data. This approach is no longer feasible as there is now many more models to evaluate. For this thesis, since the transition probability matrices are all **sparse matrices**, their adjacency matrix representations have all been converted to **adjacency list representations**. The same improvement was also made to the representation of the emission probability matrix, with the Add-One Smoothing coming into effect only during runtime.

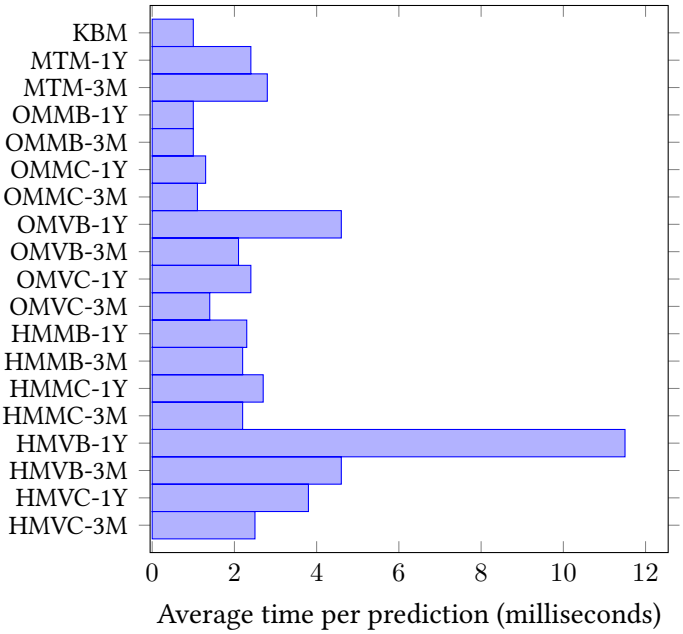

Figure 5.3: Average time per prediction taken by each model

5.6 Comparison of Prediction Accuracy

As we have considered multiple approaches in the design and training of the prediction models, we have to make the following comparisons through our experiments in order to draw useful conclusions on the decisions:

- Among the **four decoding approaches** (*i.e.* Viterbi, Constrained Viterbi, Multi-Timestep, Constrained Multi-Timestep) presented, which one is the best?
- Between the **two training approaches** (*i.e.* Year-based, Month-based) presented, which one is the best?

- Between the **two types of Markov Models** (*i.e.* Observed Markov Model, Hidden Markov Model), which one is the best?
- Among our **best-performing Markov Model and the baseline models** (*i.e.* Kinematics-Based Model, Median Trajectory Model), which one is the best?

These comparisons shall be discussed in the following sub-sections.

5.6.1 Best Decoding Approach: Constrained Multi-Timestep

Because of the large number of prediction models we have to consider, we first compare the four decoding approaches while keeping the Markov Model type and the training approach constant. The mean overall horizontal and vertical errors for each minute into the future up to the look-ahead time and for all routes are shown in Figures 5.4, 5.5, 5.6, & 5.7. From the figures, it is clear that the Multi-Timestep-based approaches are superior to the Viterbi-based approaches, regardless of the Markov Model type or the training approach, as they have consistently lower errors over all minutes into the future for all routes. The Multi-Timestep approach has slightly lower errors than the Constrained Multi-Timestep approach despite the slightly shorter prediction time it requires. However, the Constrained Multi-Timestep approach is an improvement over the Multi-Timestep approach, in that it is able to predict a trajectory where consecutive predicted positions are joined together by 1-minute transitions without incurring a considerably larger computational cost. The conclusion is that the Constrained Multi-Timestep approach is the best decoding approach to use for all four flight routes.

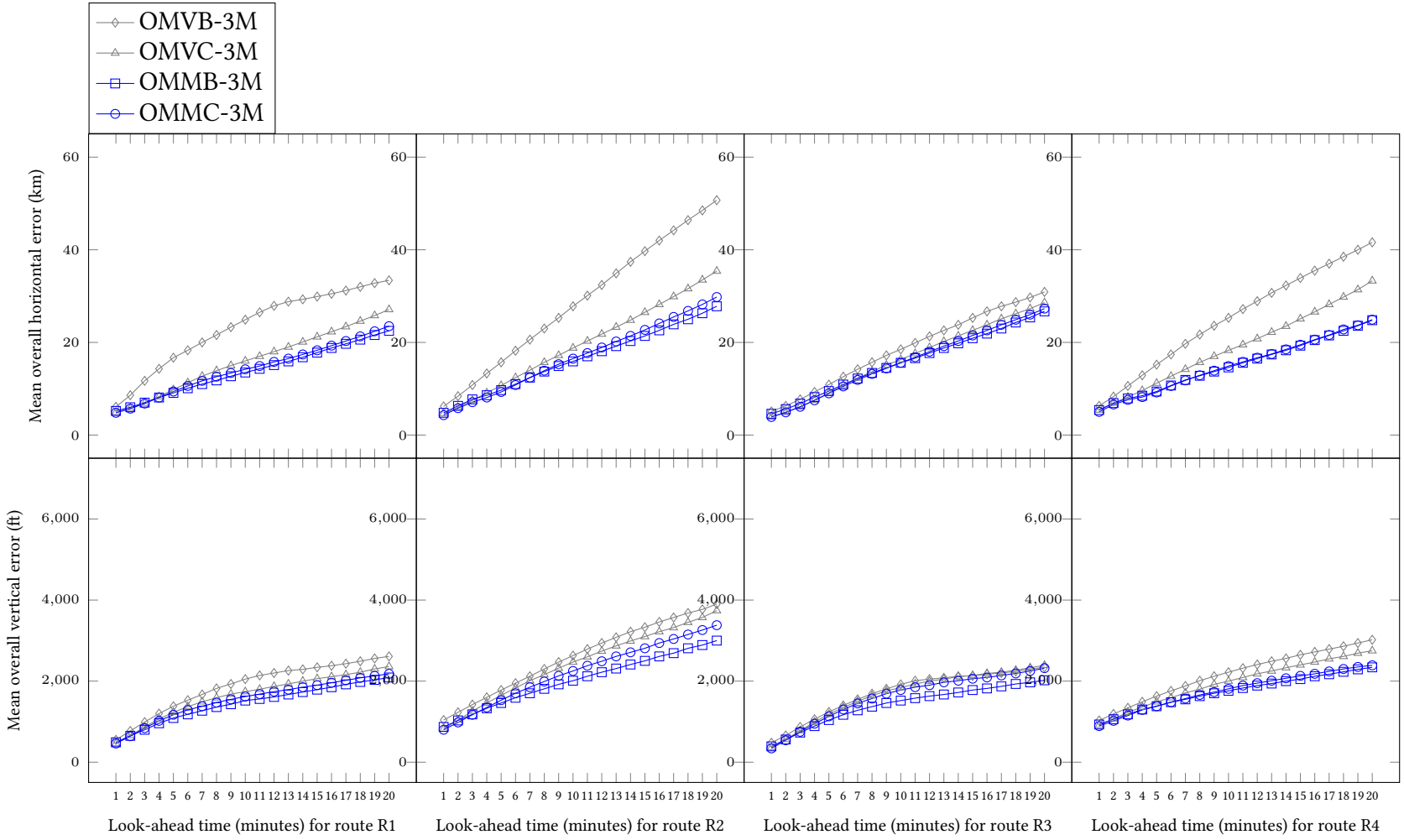

Figure 5.4: OMMs with Month-based Training for All Decoding Approaches

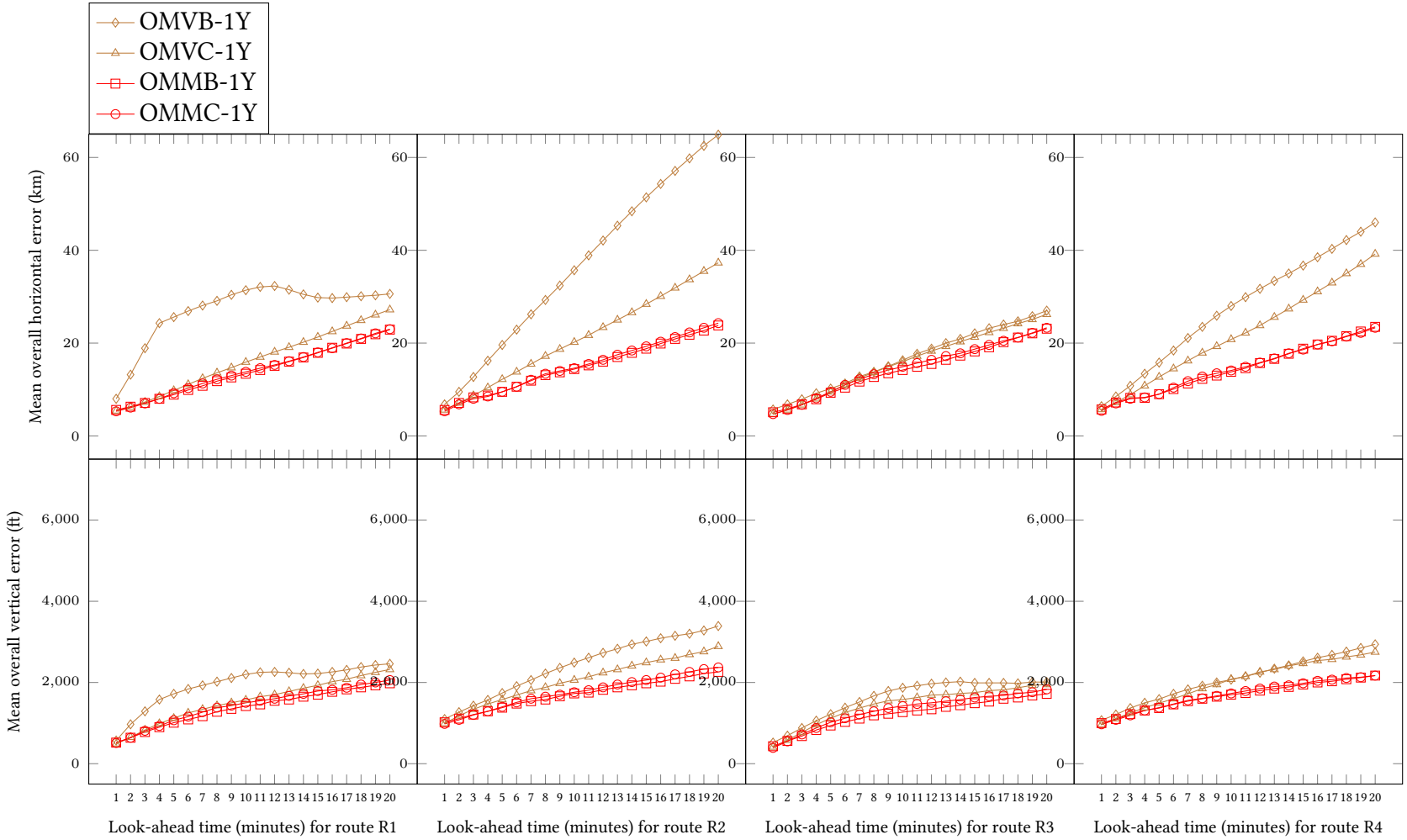

Figure 5.5: OMMs with Year-based Training for All Decoding Approaches

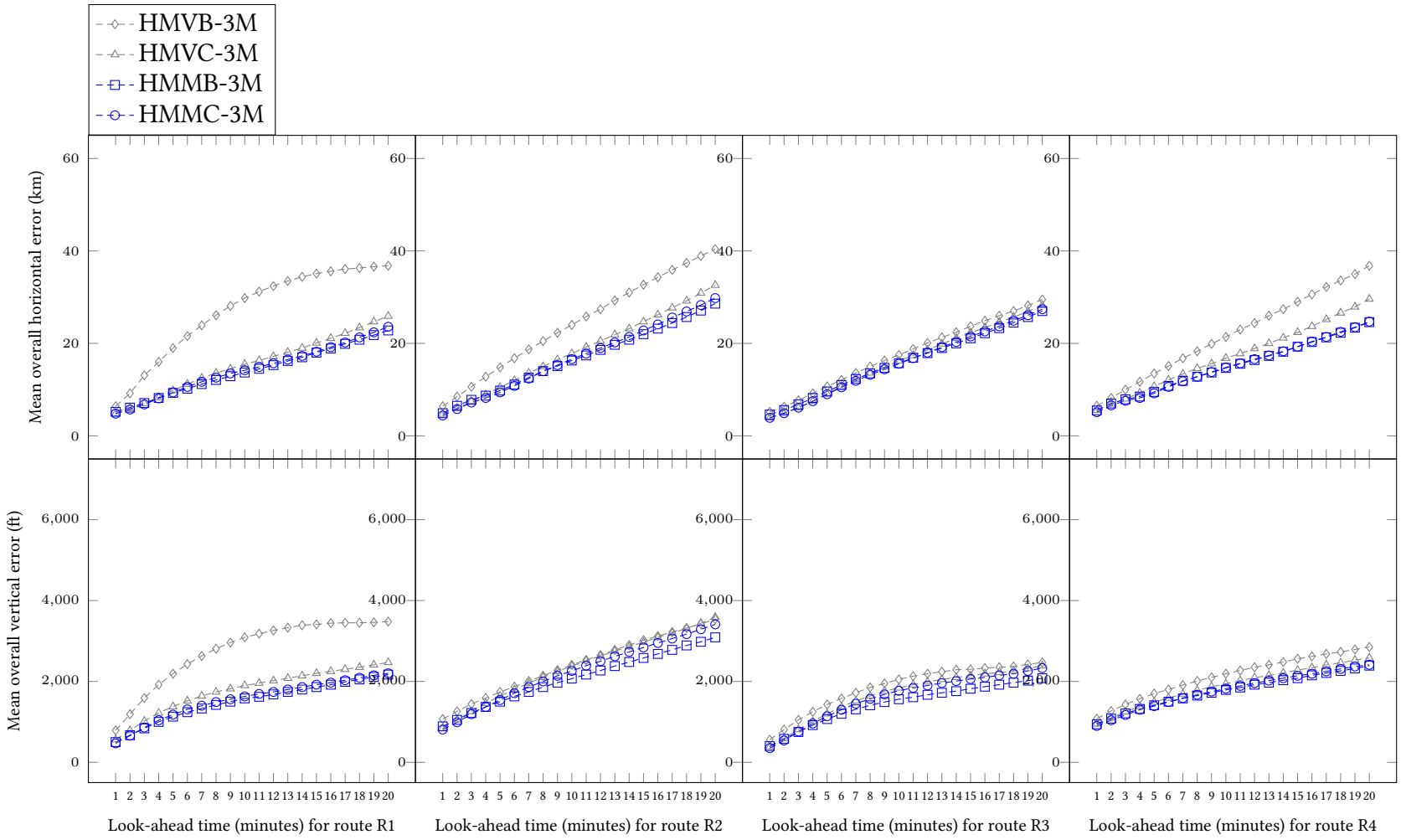

Figure 5.6: HMMs with Month-based Training for All Decoding Approaches

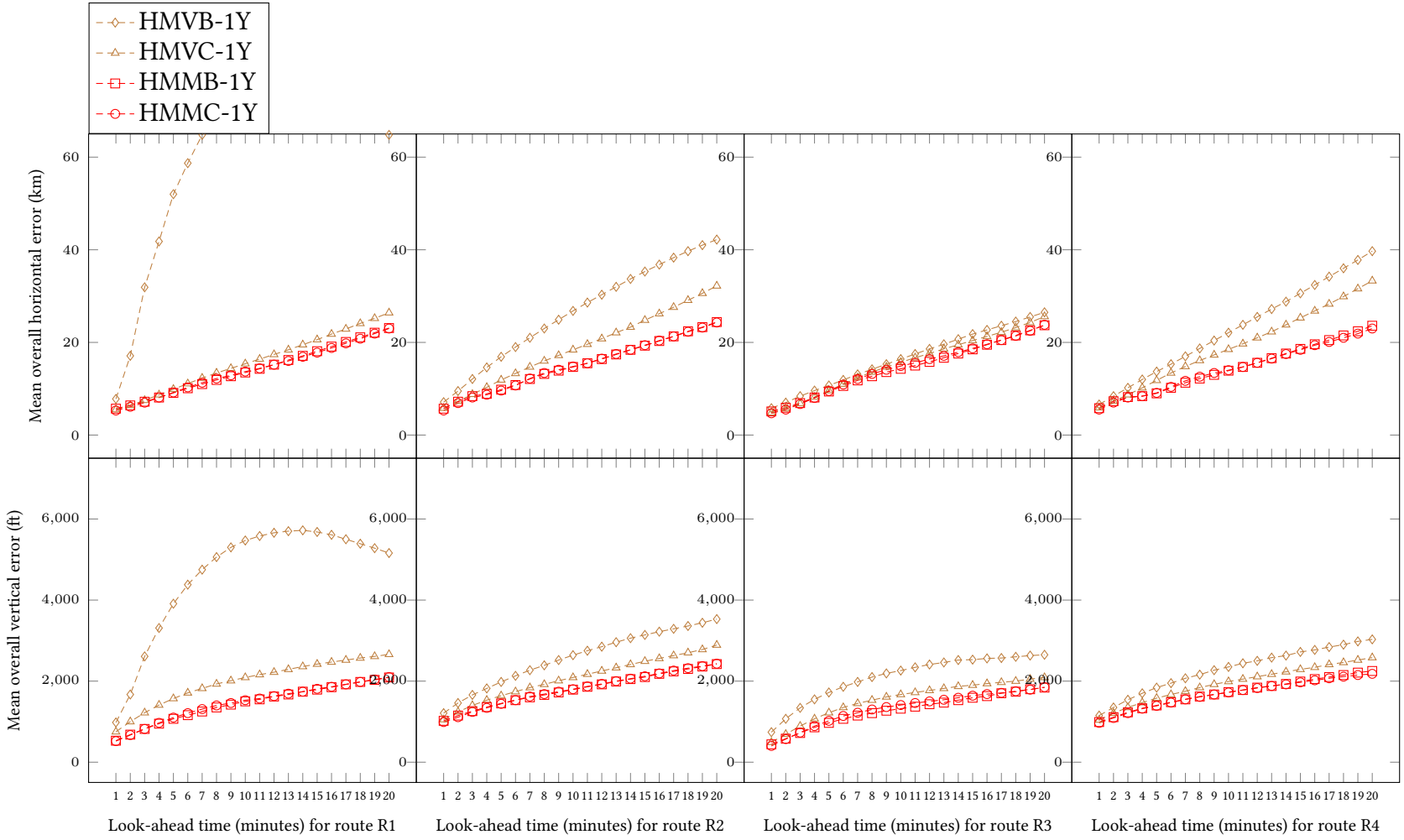

Figure 5.7: HMMs with Year-based Training for All Decoding Approaches

5.6.2 Best Training Approach: Year-based

Having decided on the Constrained Multi-Timestep decoding approach, we then consider the Markov Model type and training approach. The prediction errors for both Markov Model types and both training approaches are shown in Figure 5.8. The figure shows that regardless of the type of Markov Model, the two models trained with the Year-based approach have lower errors than the two models trained with the Month-based approach, especially for increasing minutes into the future, for routes R2, R3, & R4. This could be attributed to the amount of training data available: the Year-based approach makes use one whole year of training data, while the Month-based approach makes use of three months of training data. It also explain why their performances are closer for route R1 than the other routes: being the largest dataset, R1 provides sufficient training data for the Month-based approach. The conclusion we can draw here is that the Year-based training approach is the best training approach to use.

We are unable to obtain a better prediction accuracy for the HMMs over the OMMs despite the comparatively larger amount of effort spent on designing the HMMs and on pre-processing weather data. As the extra effort put into the HMMs have not resulted in better prediction accuracy over the OMMs, it seems that OMMs are sufficient for Trajectory Prediction for the four flight routes we have used. Nevertheless, this only shows that weather has not been a big factor in influencing the pilot's decision on where to fly for the flight routes we have used, and calls for more experiments to be conducted on other flight routes when data becomes available.

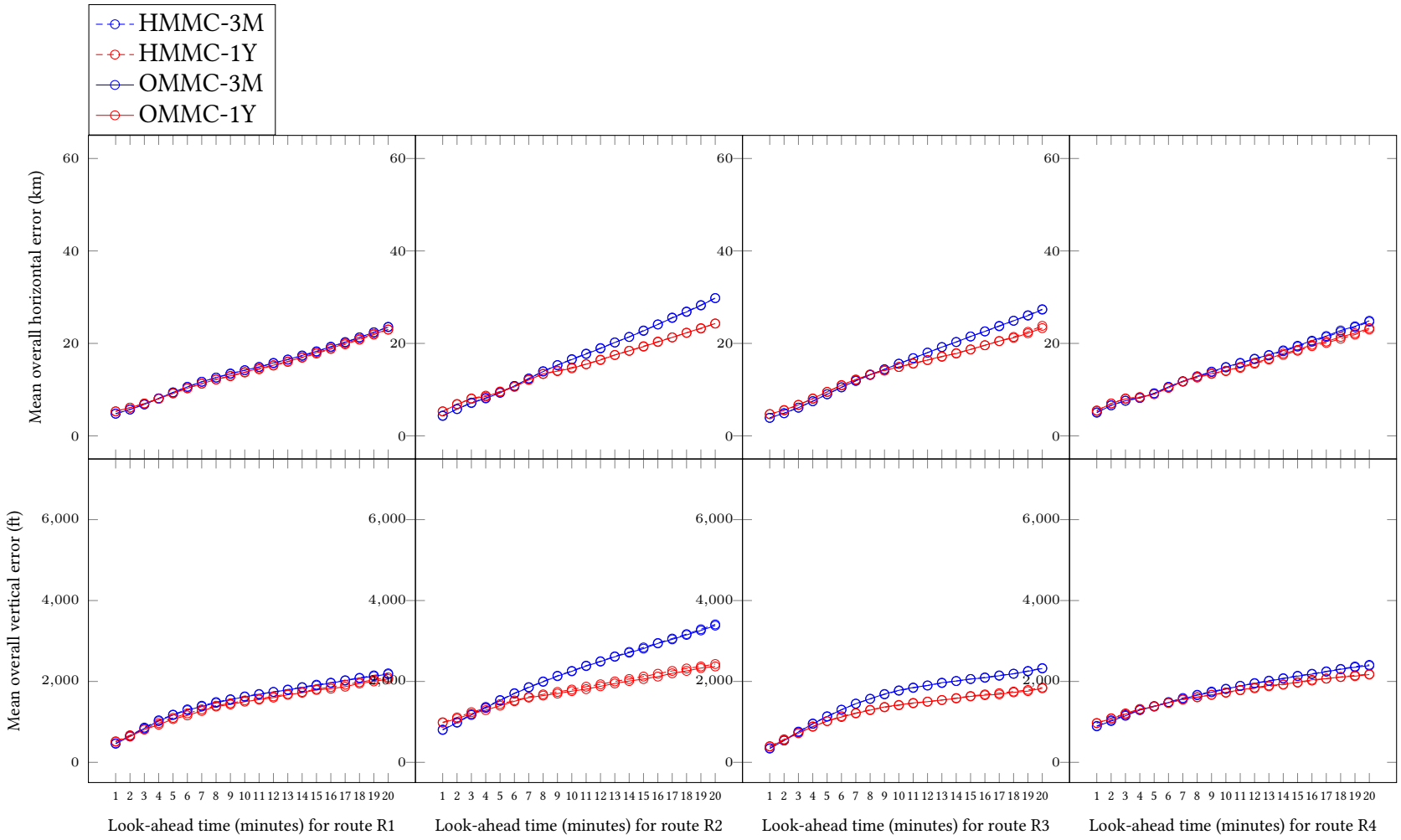

Figure 5.8: OMMs *vs.* HMMs of the Constrained Multi-Timestep Approach for Both Training Approaches

5.6.3 Comparison with Baseline Models: Markov Models are Effective

Based on the comparisons so far, the best-performing model is the Observed Markov Model with Constrained Multi-Timestep decoding and Year-based training (*i.e.* OMMC-1Y), which we shall use to understand how well our Markov Models can perform when compared with traditional approaches. The prediction accuracy of this model and of the baseline models (*i.e.* KBM, MTM-1Y, MTM-3M) are shown in Figure 5.9. It can be seen that model OMMC-1Y is able to make more accurate predictions than the baseline models for both horizontal and vertical dimensions, especially for the 10^{th} minute onwards. While the baseline model KBM is able to make highly accurate predictions for up the 3^{rd} minute, its prediction accuracy drops rapidly with increasing time from prediction due to the increasing uncertainty in kinematics parameters.

The errors for baseline models MTM-1Y & MTM-3M are generally larger than model OMMC-1Y, with no clear winner among the two training approaches for the MTM. It seems that they are able to perform equally well with the model OMMC-1Y near the look-ahead time of 20 minutes, as seen from their errors converging for the vertical dimension of route R2 and both horizontal and vertical dimensions of route R4. To verify whether this is correct, we study the mean and standard deviation of intra-month errors at the look-ahead time of 20 minutes, as shown in Figures 5.10 & 5.11. Contrary to what we might conclude from looking at Figure 5.9 alone, Figures 5.10 & 5.11 show that the models MTM-1Y & MTM-3M have larger standard deviations for intra-month errors than model OMMC-1Y most of the time, which indicate inconsistent performance. This inconsistency is due to the tendency of a MTM model to select the trajectory that represents the shortest and most direct route between the origin and destination airports, resulting in very accurate predictions for the bulk of test trajectories, but extremely bad predictions for some trajectories that take a longer route due to external factors, such as congestion at the destination airport.

We also present the mean and standard deviation of overall errors at the look-ahead time of 20 minutes in Figure 5.12. The baseline model KBM has the best

performance for route R3 and the worst for route R2, which is because route R3 has the longest mean flight time among all routes, while route R2 has the shortest mean flight time, as seen in Table 5.1. Flights on a longer route spend a larger proportion of the flight time cruising in a straight line towards the destination airport, allowing the KBM to make accurate predictions because there is little change in speed of the flight trajectory for these large portions of the flights. The baseline models MTM-1Y & MTM-3M have the best performance for route R4 compared to other routes, as seen from the low means and low standard deviations of both horizontal and vertical errors for route R4. This is due to route R4 having the smallest set of trajectories among all routes, as seen in Table 5.1, and thus having less variability among trajectories compared to other routes, as having more varied trajectories causes a MTM model to be unable to make accurate predictions. The model OMMC-1Y, on the contrary, is able to perform well on all flight routes, especially when a large amount of training data is available.

We can thus conclude that our proposed model OMMC-1Y outperforms, in general, the baseline models significantly, except when making predictions for up to the first few minutes, for which the KBM approach seems more appropriate. A KBM approach, however, is in general not appropriate for look-ahead times larger than a few minutes, giving predictions with rapidly increasing errors as the look-ahead time increases, even though the *average* prediction accuracy of KBM improves with increasing mean flight time because longer flights cruise for a longer time on a straight trajectory with more or less constant speed. The MTM approach also performs overall significantly worse than our proposed model, and is typically even outperformed by the KBM approach for up to intermediate look-ahead times. It is clear that the larger the training set, the better the performance of our proposed Markov Model approach, and we therefore expect that the performance gap between our proposed model and the baseline models to increase as more and more historical flight data becomes available.

As demonstrated for the look-ahead time of 20 minutes for route R1, which has the largest dataset available for training, our proposed model OMMC-1Y achieved a mean overall horizontal error of 23.0 km and mean overall vertical error of

2060 ft. In the horizontal dimension, the error of 23.0 km is approximately 1.70 times of a grid cube away (having width of 13.545 km), but is small when compared to an aircraft being able to travel 219 km in most cases (*i.e.* median) and 264 km in extreme cases (*i.e.* 90$^{\text{th}}$ percentile) within 20 minutes. Looking at the vertical dimension, the error of 2060 ft is approximately 1.03 times of a vertical interval away (having a separation of 2000 ft), but is small when compared to an aircraft being able to ascend or descend 15500 ft in most cases (*i.e.* median) and 26800 ft in extreme cases (*i.e.* 90$^{\text{th}}$ percentile) within 20 minutes. The errors represent an improvement of 24.6% in horizontal error and 34.2% in vertical error for route R1 over MTM-3M, the best-performing baseline model.

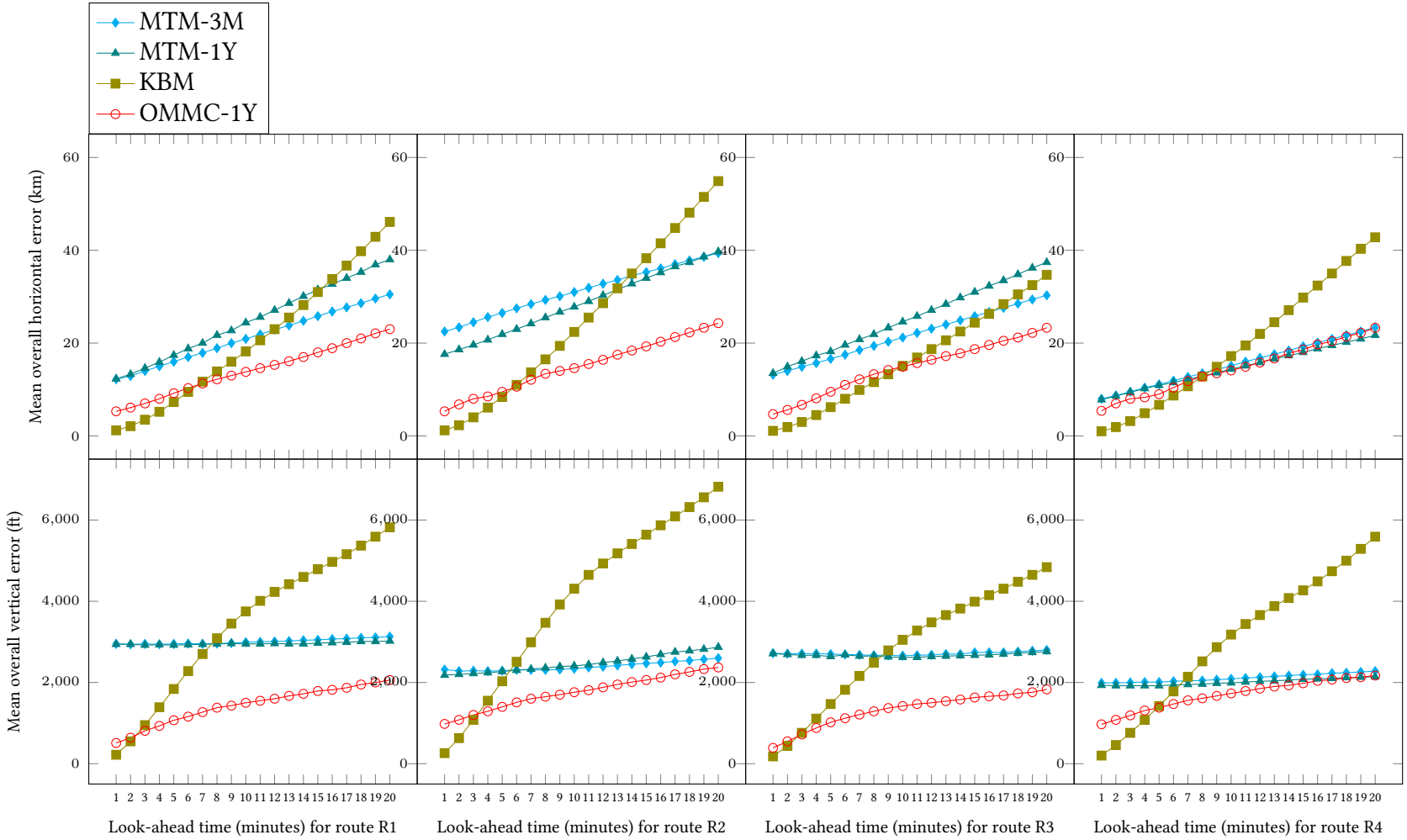

Figure 5.9: The Best-Performing Model *vs.* the Baseline Models

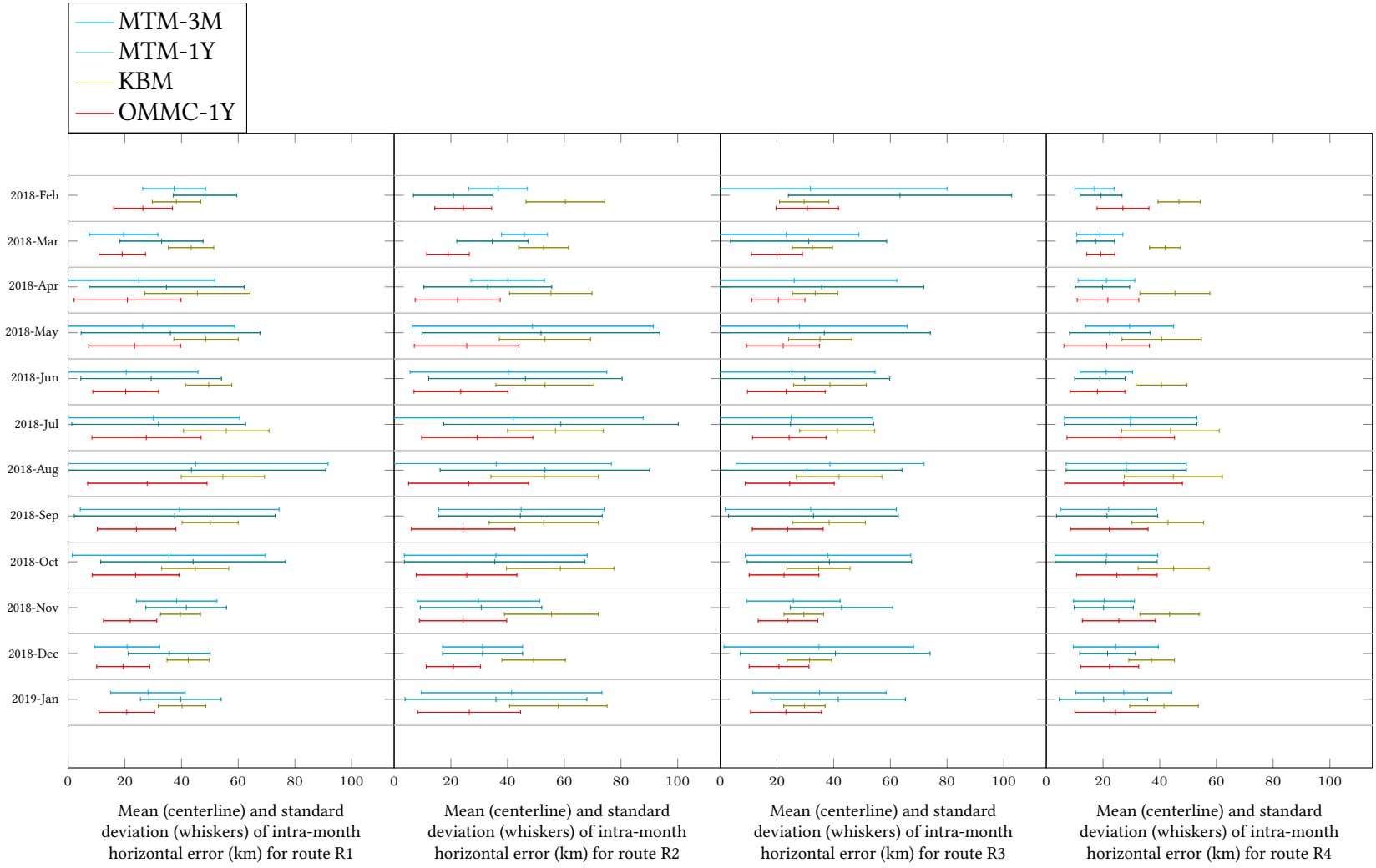

Figure 5.10: Mean and standard deviation of intra-month horizontal error at look-ahead time of 20 minutes

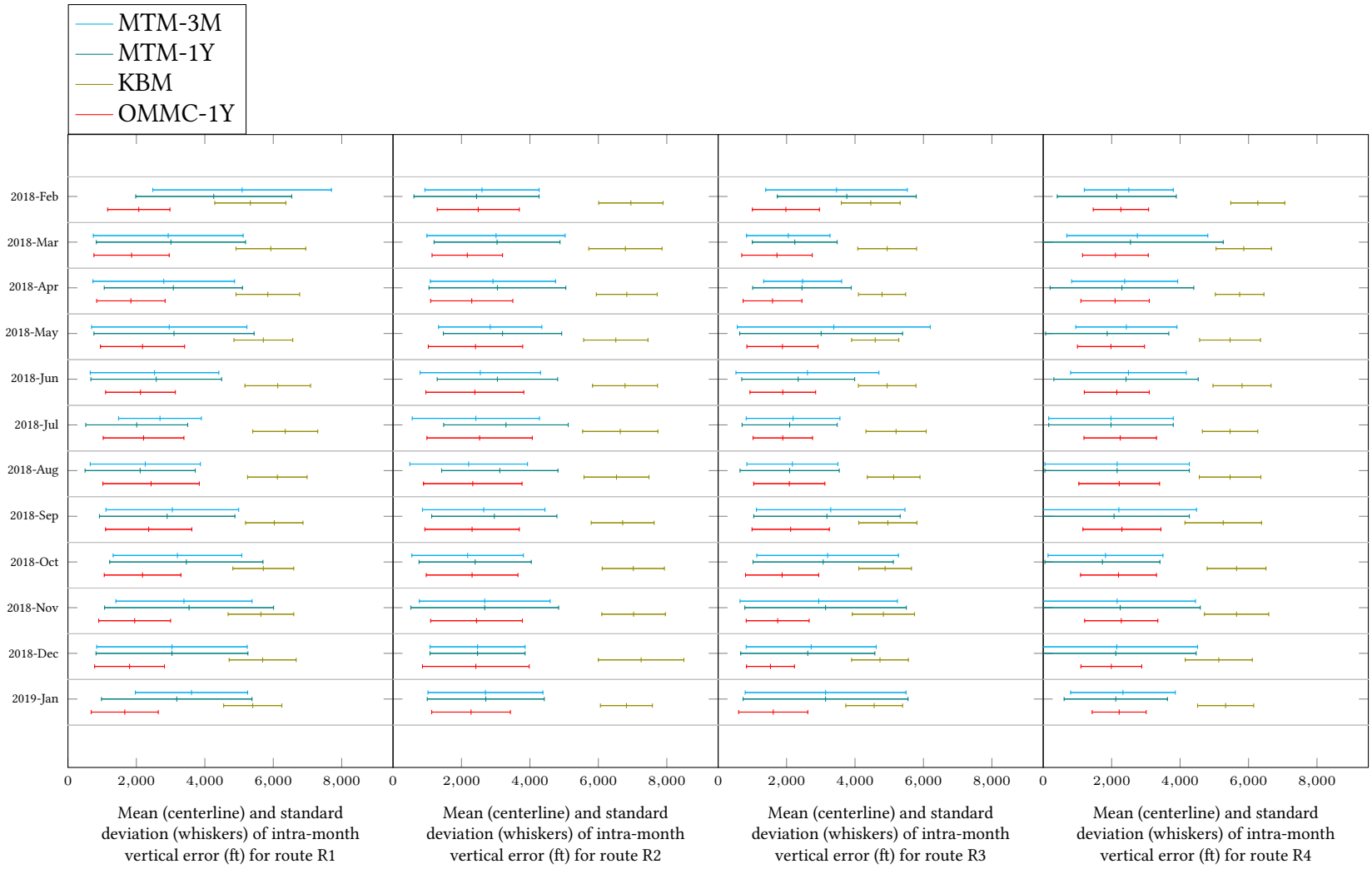

Figure 5.11: Mean and standard deviation of intra-month vertical error at look-ahead time of 20 minutes

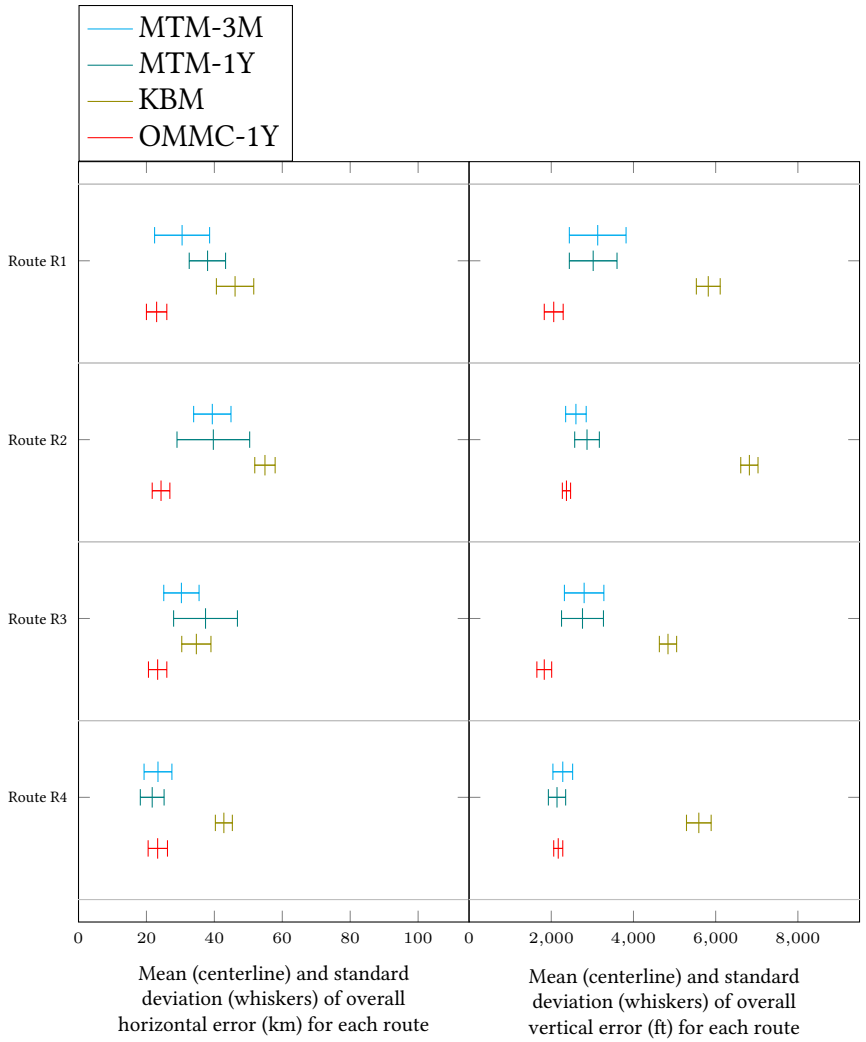

Figure 5.12: Mean and standard deviation of overall horizontal and vertical errors at look-ahead time of 20 minutes

5.7 Using Different Weather Bin Intervals

One conclusion we have drawn from the previous section is that the HMMs we have presented were unable to make more accurate predictions that their OMM equivalents. This prompted us to re-think our approach for incorporating weather information, and in particular, the way we have discretized weather vectors. The conjecture that we based our decision on is that even though a flight plan has been filed before the actual flight takes place, there are sometimes unexpected changes in local weather that warrant a change in the flight path. Because of this, it is important that we discretize weather vectors representing extreme weather conditions into different bin vectors from those representing common weather conditions. We perform additional experiments by using the 2^{nd} and 3^{rd} standard deviations from the mean in determining the bin 'split', and compare them with that using the 1^{st} standard deviation in Table 5.2, and with the OMM equivalents. Based on the four decoding approaches and two training approaches, we have a total of eight comparisons, as shown in Figures 5.13 to 5.20, where a particular HMM (*e.g.* HMVB-3M) is compared with its OMM equivalent (*e.g.* OMVB-3M) and the additional HMM models (*e.g.* H2VB-3M and H3VB-3M, representing models having the bin 'split' at the 2^{nd} and 3^{rd} standard deviations, respectively).

From Figures 5.13 & 5.14, it seems that for the Viterbi approach, the HMM models achieved mixed results, but all three HMM models performed worse than the OMM model on route R1, the largest dataset among the four routes. As for each of the other six, Figures 5.15, 5.16, 5.17, 5.18, 5.19, & 5.20, the four models being compared in each figure have almost the same prediction accuracies. In particular, for the Constrained Multi-Timestep approach in Figures 5.19 & 5.20, the four models are almost non-distinguishable from one another. This shows the effectiveness of using multi-timestep transition probabilities in the Markov Model, as it is able to limit the number of states being considered at each minute to only those that have occurred in historical trajectories. Given how effective this approach has been, we were unable to improve the prediction accuracy further by incorporating local weather information through the use of HMMs.

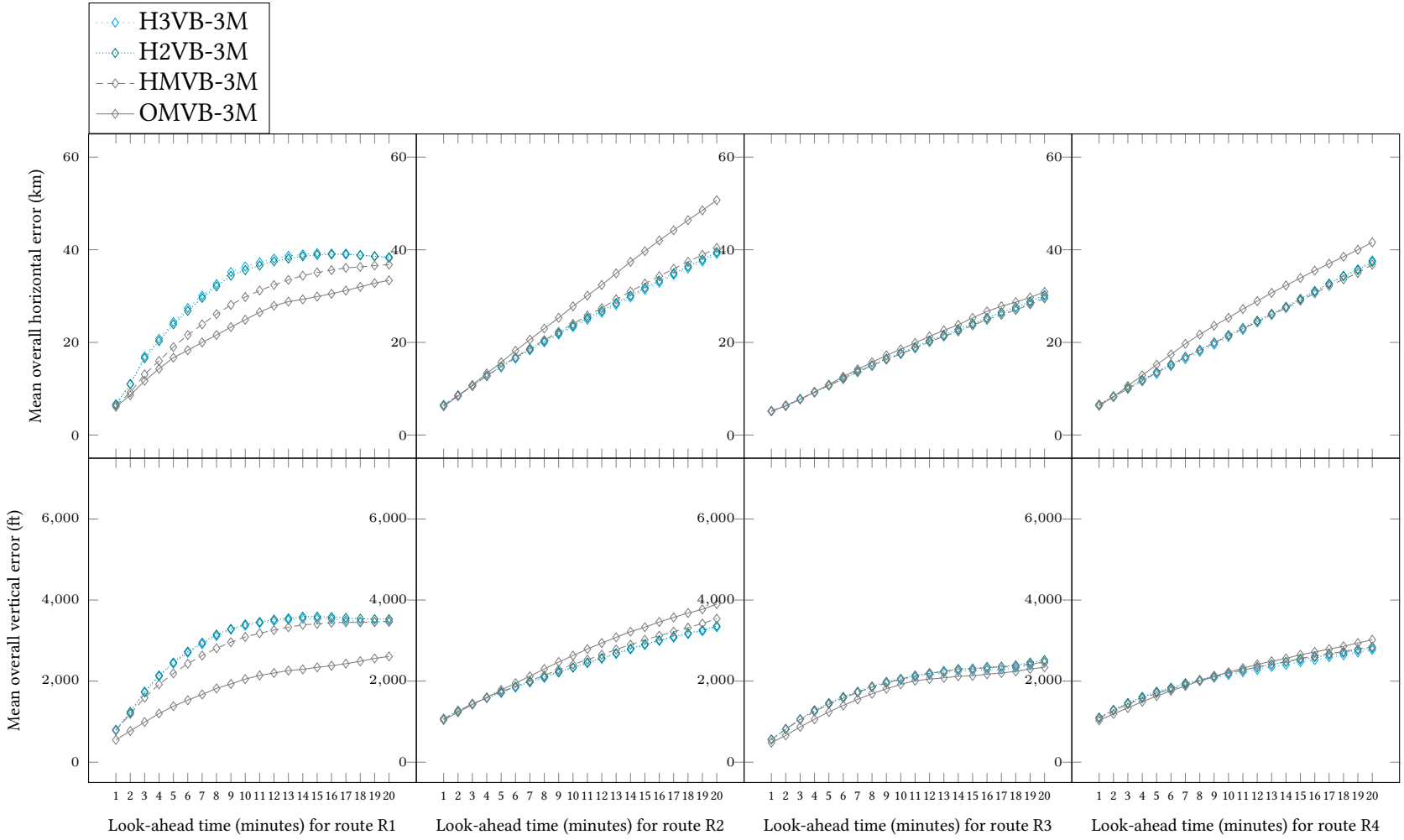

Figure 5.13: OMM *vs.* HMMs of Different Bin Splits of the Viterbi Approach with Month-based Training

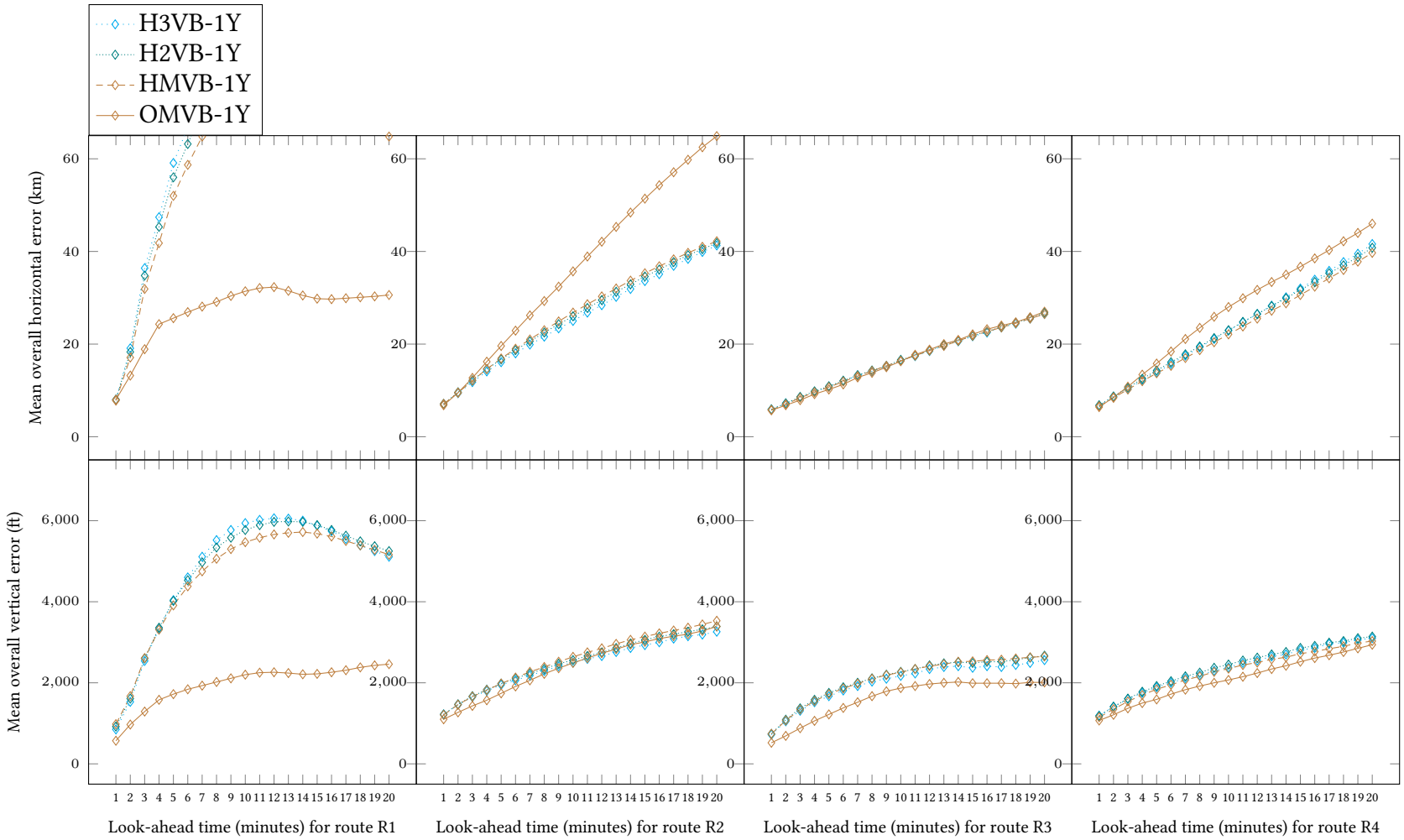

Figure 5.14: OMM *vs.* HMMs of Different Bin Splits of the Viterbi Approach with Year-based Training

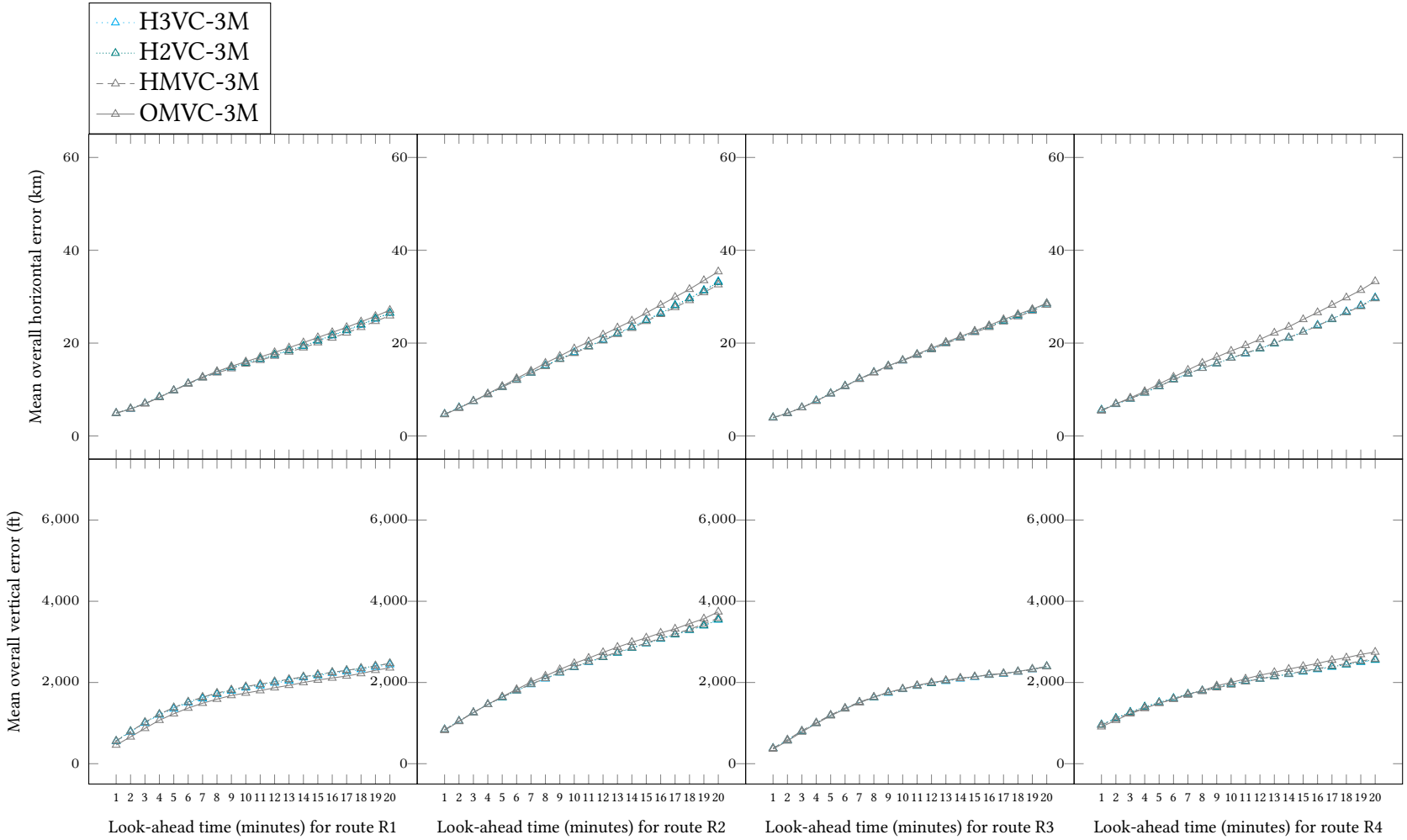

Figure 5.15: OMM *vs.* HMMs of Different Bin Splits of the Constrained Viterbi Approach with Month-based Training

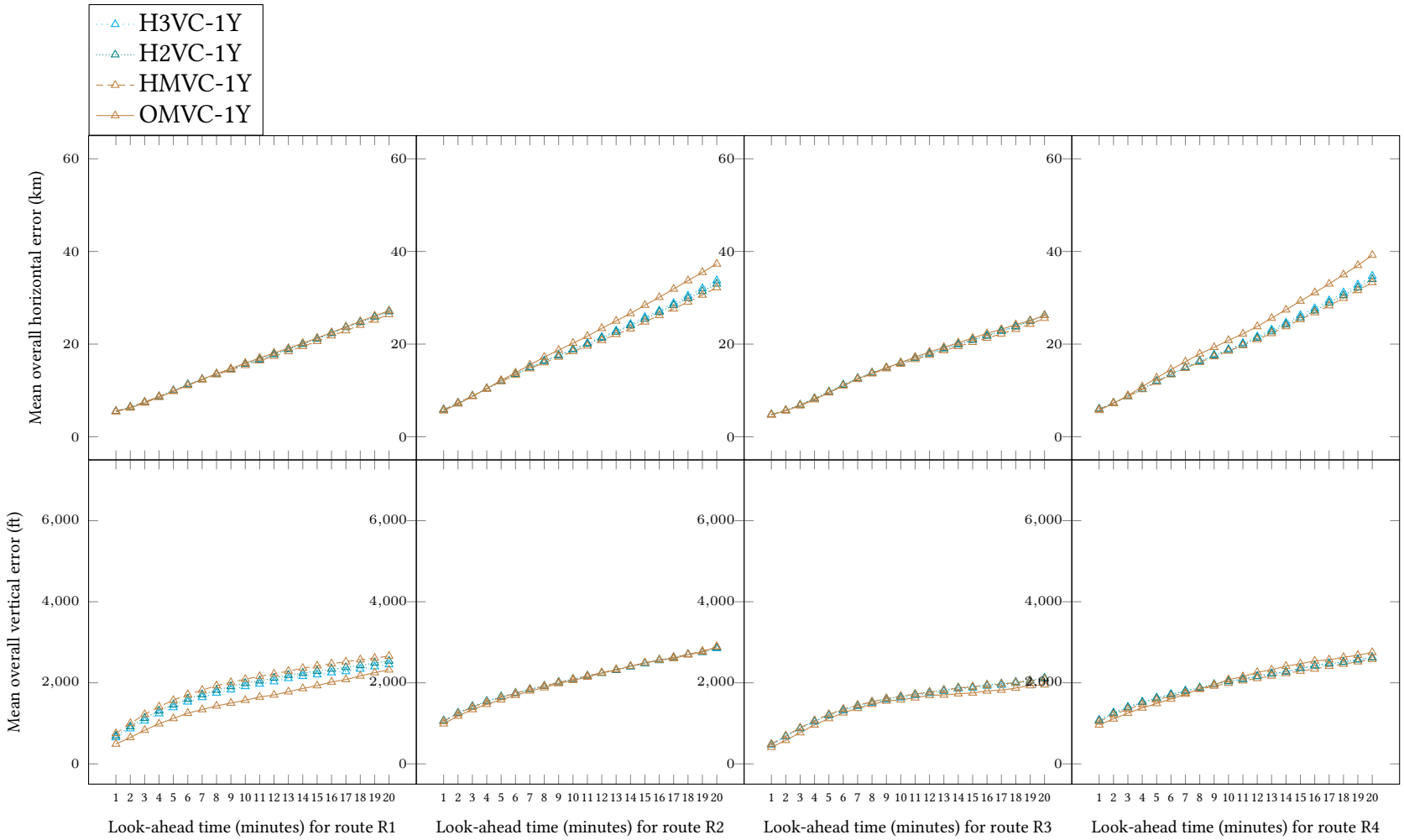

Figure 5.16: OMM *vs.* HMMs of Different Bin Splits of the Constrained Viterbi Approach with Year-based Training

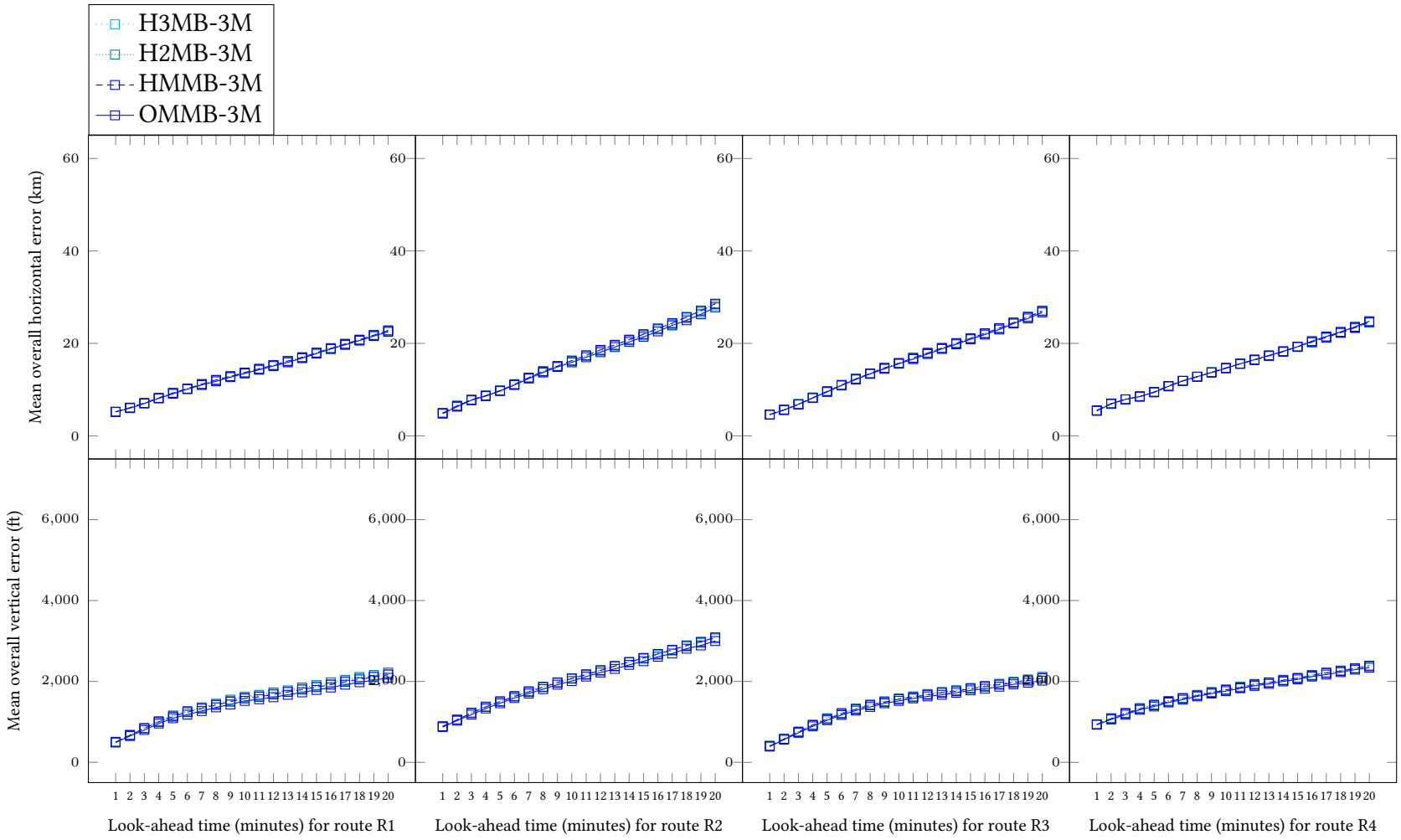

Figure 5.17: OMM *vs.* HMMs of Different Bin Splits of the Multi-Timestep Approach with Month-based Training

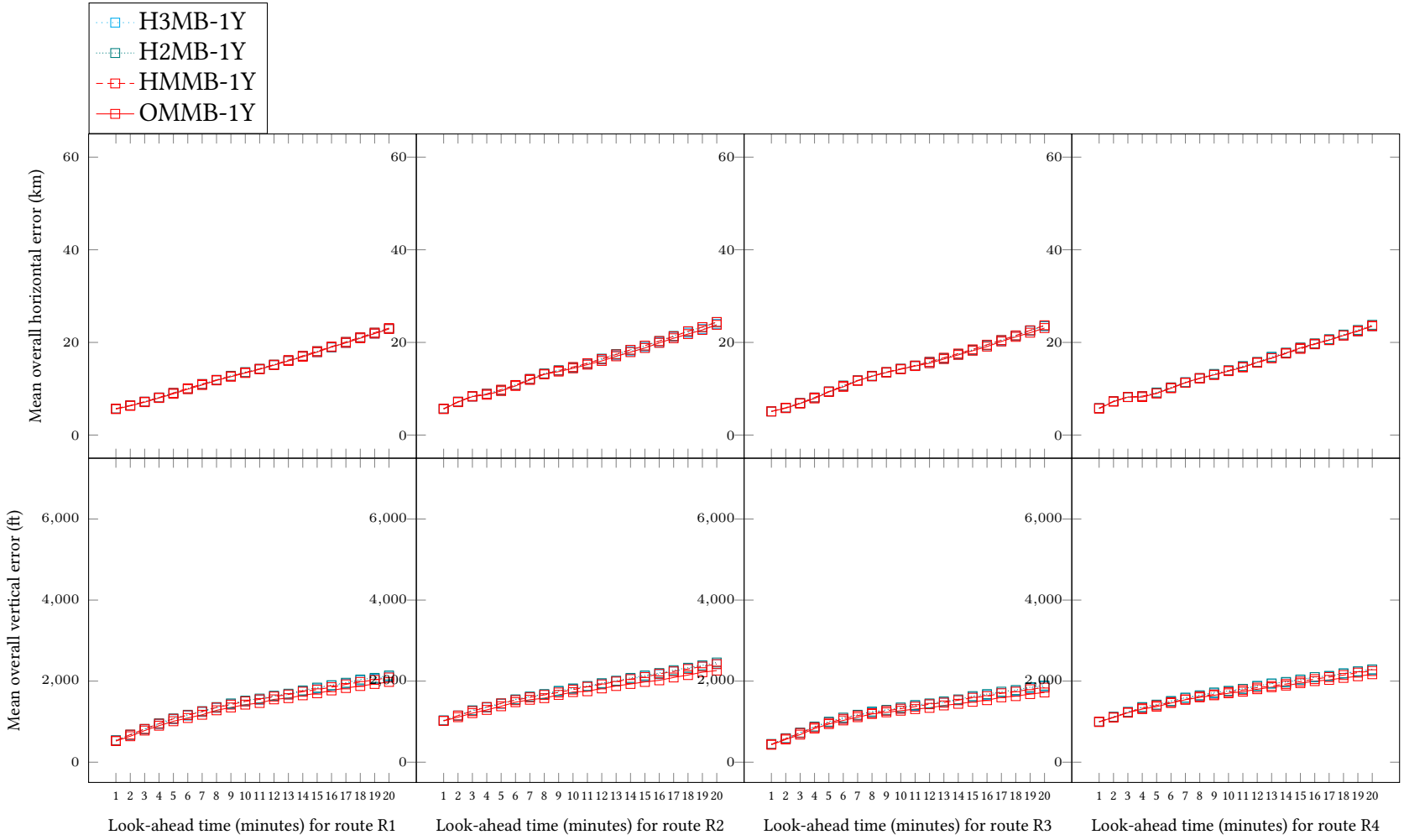

Figure 5.18: OMM *vs.* HMMs of Different Bin Splits of the Multi-Timestep Approach with Year-based Training

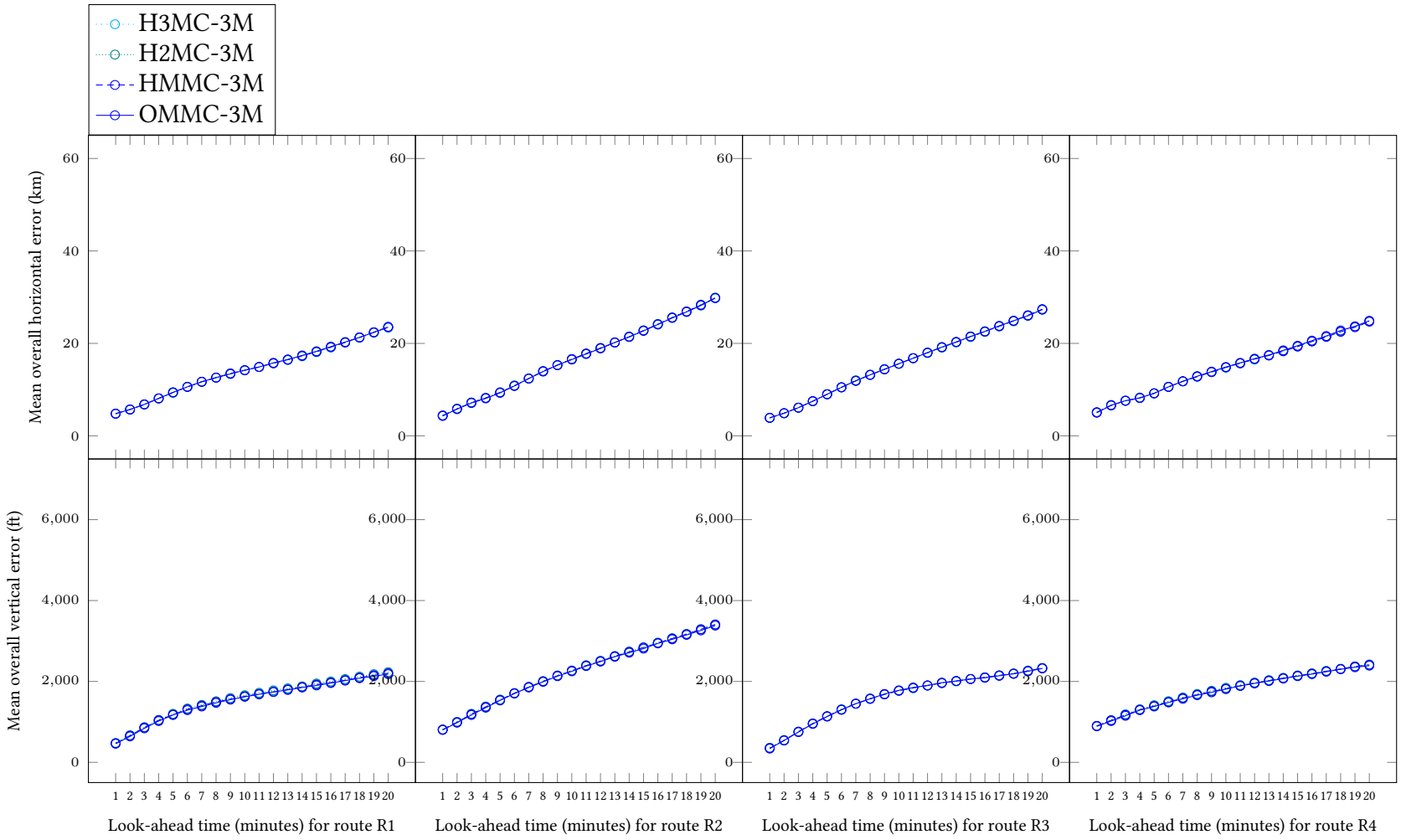

Figure 5.19: OMM *vs.* HMMs of Different Bin Splits of the Constrained Multi-Timestep Approach with Month-based Training

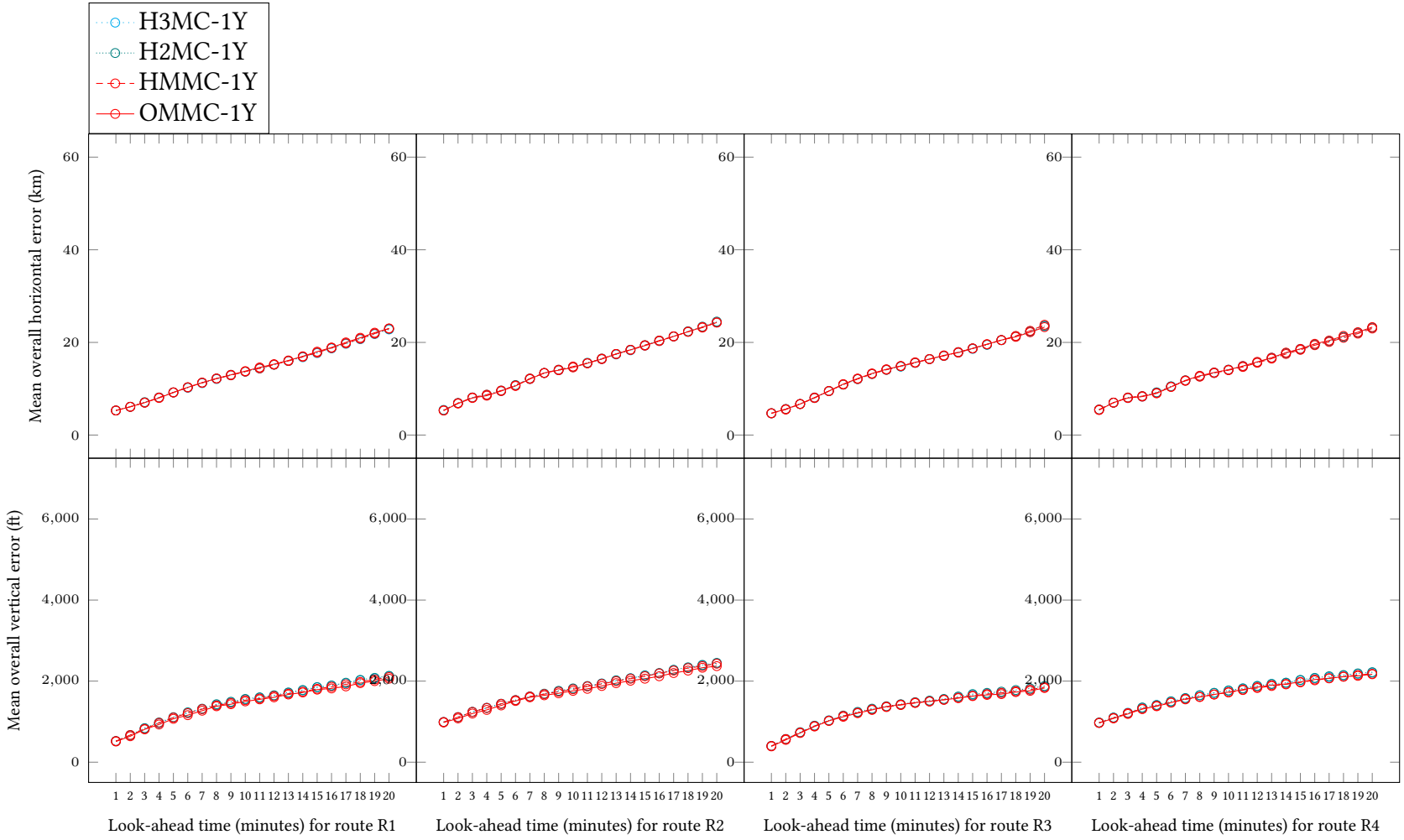

Figure 5.20: OMM *vs.* HMMs of Different Bin Splits of the Constrained Multi-Timestep Approach with Year-based Training

5.8 Reducing Unsuccessful Predictions

Our proposed Markov Models are unable to make predictions for some test positions, and we deal with such cases by falling back on the KBM model to make predictions so that a fair comparison can be made across all proposed and baseline models. Specifically, this happens when given a test position $pos_{ts_{now}}$ that maps to start state $q_{ts_{now}}$, a Markov Model might have a zero transition probability out of the start state $q_{ts_{now}} = s_i \in S$ for a particular timestep t, *i.e.* $\sum_{s_j \in S} a_t(q_{ts_{now}}, s_j) = \sum_{s_j \in S} a_t(s_i, s_j) = 0$. This happens when none of the historical trajectories in the training set has ever flown into grid cube gc_i corresponding to state s_i, resulting in $a_t(s_i, s_j) = 0$ when Equation 4.1 is applied (actually it results in an error because the denominator is zero, but because such cases occurs very often, for practical purposes they are ignored by assuming zero probability). When we analyze the Constrained Multi-Timestep models (*i.e.* models based on the best decoding approach), we found that there is a large number of such occurrences, which we call **unsuccessful predictions**, as shown in Table 5.3. They happen because of the strict constaints imposed by Equation 4.6 requiring non-zero transition probabilities for all timesteps $t \in [1, L]$ out of the start state, and requiring the states for each minute to have non-zero 1-minute transition probabilities into states for the next minute.

To mitigate this problem, we tried to do the following for unsuccessful predictions:

- We run the prediction again, this time replacing the start state with a set of states formed by the **neighbours** of the start state, where we define two states to be neighbours when their corresponding grid cube indices in our 3-dimensional reference grid have a difference of 1 or less along all 3 dimensions. Thus, each state has $3^3 = 27$ neighbouring states, including itself, and we use $\Gamma(s_i)$ to denote the set of states that are neighbours of state s_i.
- We replace the initialization clause of the Viterbi algorithm with Equation 5.1, such that in the first minute into the future, we assign a Viterbi probability to each state by taking the maximum of the 1-minute transition probabilities

from all neighbouring states of the start state.

- We do the same for the recursion clause in Equation 4.6 (for the Hidden Markov Model based on the Constrained Multi-Timestep Approach), replacing it with Equation 5.2.

$$vt_1(s_j) = b(s_j, o_{j,t}) \times \max_{s_l \in \Gamma(q_{ts_{now}})} \big(a_1(s_l, s_j)\big), \forall s_j \in S. \tag{5.1}$$

$$vt_t(s_j) = b(s_j, o_{j,t}) \times \max_{s_l \in \Gamma(q_{ts_{now}})} \big(a_t(s_l, s_j)\big) \times \max_{s_i \in S} \big(vt_{t-1}(s_i) \times sgn\big(a_1(s_i, s_j)\big)\big), \\ \forall s_j \in S, \forall t \in [2, L]. \tag{5.2}$$

As a result of this change (which we call **start state expansion**), we were able to greatly reduce the number of unsuccesful predictions, as shown in Table 5.3, such that the best performing model, OMMC-1Y, only has 10% of predictions being unsuccessful for route R3, and 3.8% or less for the other routes R1, R2, & R3.

We also examined the prediction accuracy resulting from this change, as shown in Figures 5.21 & 5.22, where the code of a model that has gone through start state expansion is appended with a '+' suffix, *i.e.* the modified version of OMMC-1Y is OMMC-1Y+. From the figures, it is obvious that the prediction accuracy of the models has been improved for larger look-ahead times, though prediction accuracy for the first few minutes has dropped due to the uncertainty introduced through start state expansion. The models with a Month-based training approach have seen the greatest improvement, because they have had the largest proportions of unsucessful predictions being reduced (by as much as 20% for Route R3). Also, the prediction accuracy for Route R1 has had the least improvement. This is because Route R1 is the largest dataset, and already has a sufficient amount of training data available even when the Month-based training approach is used. Based on these findings, we believe that this is a reasonable approach in handling unsuccessful predictions by the proposed Markov Models, and is worth exploring further in future experiments.

Table 5.3: Number of unsuccessful predictions before and after the change

Model	Number of predictions by KBM (before change)	Number of predictions by KBM (after change)
Total number of predictions for Route R1: 514,116		
OMMC-3M / HMMC-3M	120,200 (23.4%)	37,501 (7.3%)
OMMC-1Y / HMMC-1Y	69,166 (13.5%)	11,803 (2.3%)
Total number of predictions for Route R2: 223,055		
OMMC-3M / HMMC-3M	75,631 (33.9%)	33,214 (14.9%)
OMMC-1Y / HMMC-1Y	39,389 (17.7%)	5,353 (2.4%)
Total number of predictions for Route R3: 271,291		
OMMC-3M / HMMC-3M	105,815 (39.0%)	51,503 (19.0%)
OMMC-1Y / HMMC-1Y	61,681 (22.7%)	27,001 (10.0%)
Total number of predictions for Route R4: 132,047		
OMMC-3M / HMMC-3M	32,618 (24.7%)	13,785 (10.4%)
OMMC-1Y / HMMC-1Y	22,295 (16.9%)	5,041 (3.8%)

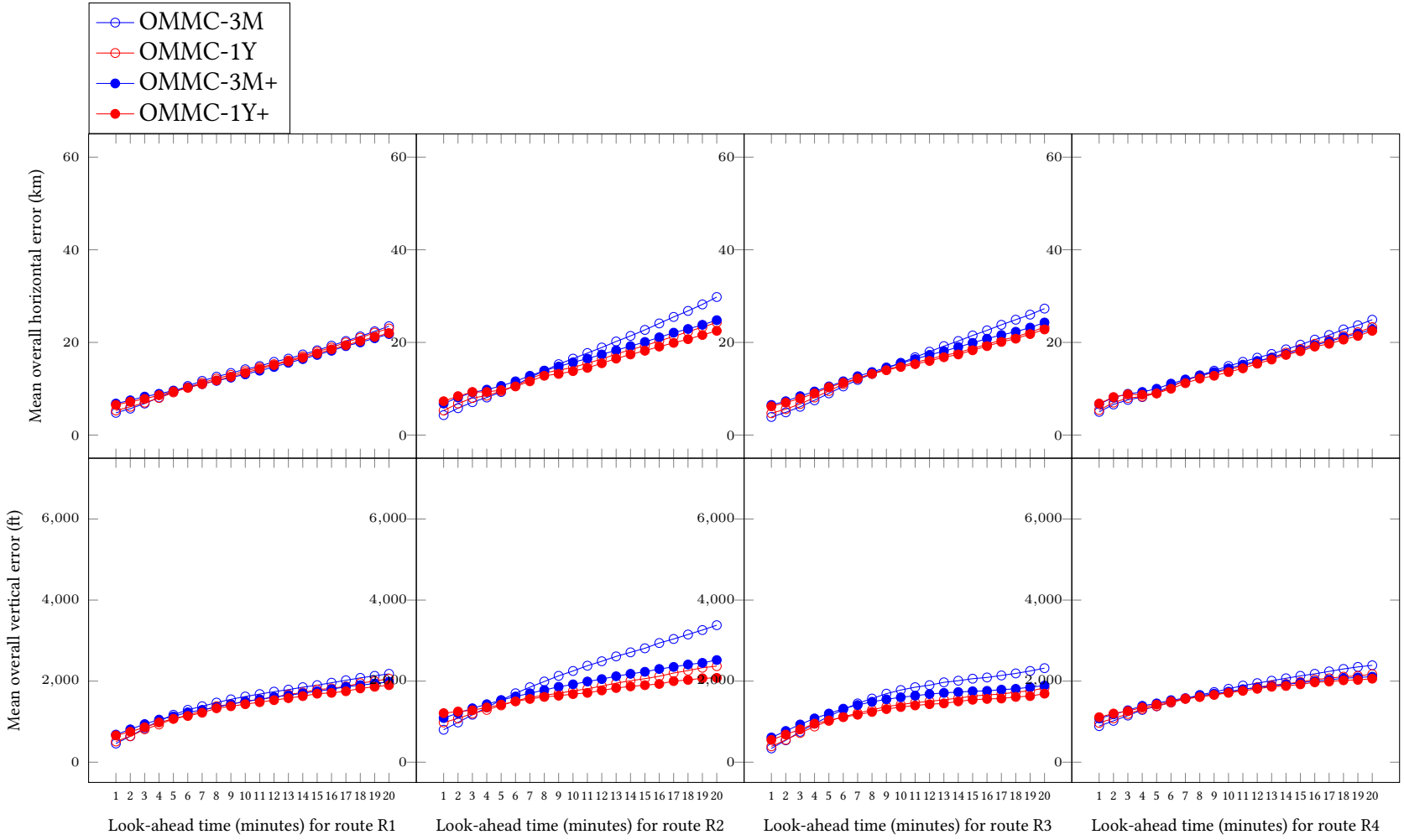

Figure 5.21: OMM *vs.* OMM+ of the Constrained Multi-Timestep Approach for Both Training Approaches

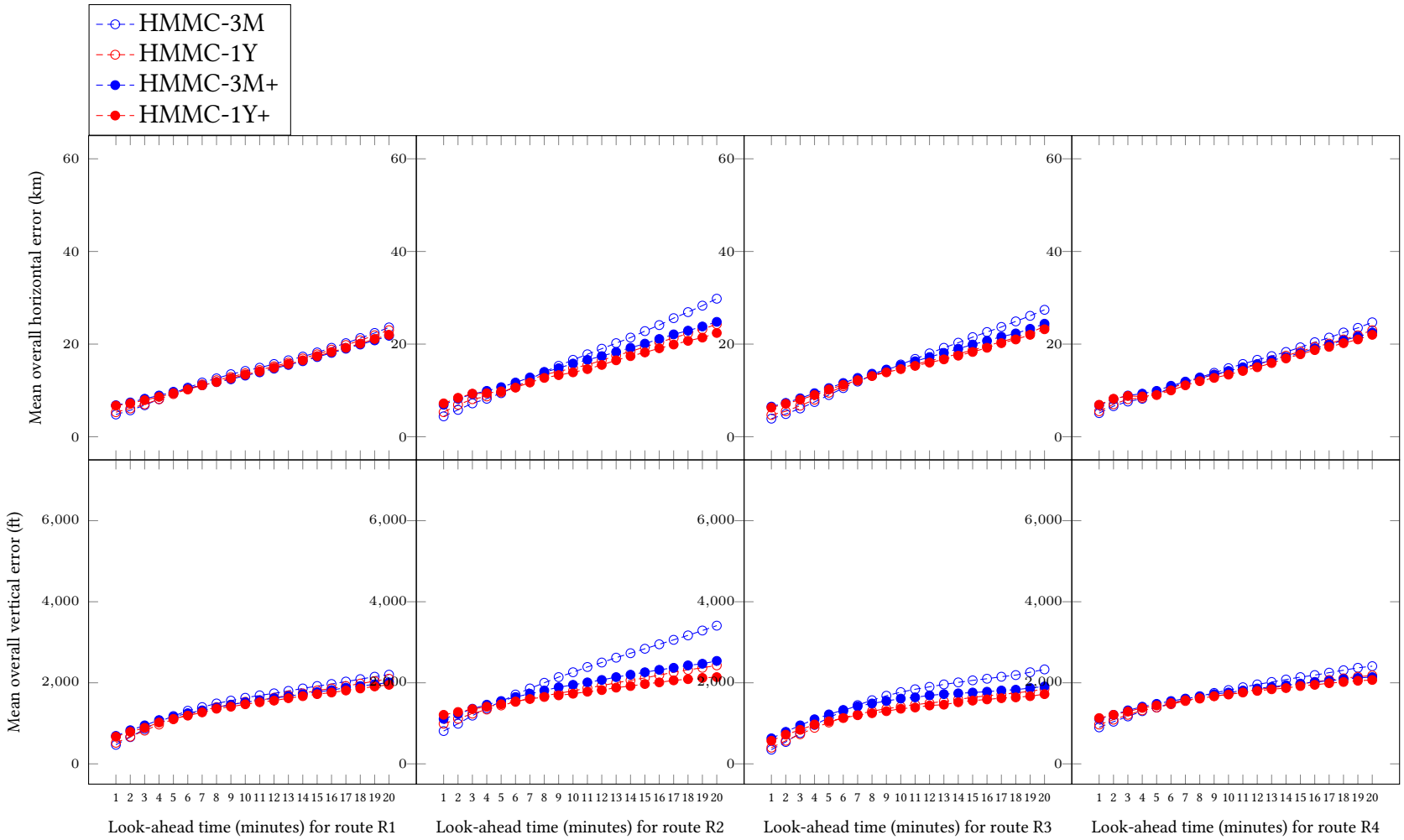

Figure 5.22: HMM *vs.* HMM+ of the Constrained Multi-Timestep Approach for Both Training Approaches

Chapter 6

Conclusion

In this thesis, we have presented a statistical learning approach to the online prediction of mid-flight aircraft trajectories. Through the mapping of the problem onto Hidden Markov Models, we were able to incorporate local weather information for trajectory prediction. We used a rich dataset of historical trajectories and performed extensive experiments for comparing multiple prediction models of various configurations. Using prediction accuracy metrics for both horizontal and vertical errors, we demonstrated that we were able to achieve better prediction accuracy when compared to conventional approaches, especially when we need to predict trajectories further into the future, while not requiring more time to make predictions (all of the work was done on an off-the-shelf laptop computer).

6.1 Future Work

Firstly, for our models following the Constrained Multi-Timestep decoding approach, we were unable to achieve a better prediction accuracy by a Hidden Markov Model (HMM) over an Observed Markov Model (OMM) equivalent. We believe this is because we have already achieved a very level of accuracy through the use of multi-timestep transition probabilities, though it is also important to perform experiments on flight routes that go over large bodies of water, such as the North Atlantic Tracks, when data for such flight routes becomes available. It will also be interesting to explore other ways of incorporating weather information, or ways of incorporating other information about the flight in the future.

Secondly, we pointed out that there is a large number of unsuccessful predictions for our proposed Constrained Multi-Timestep Markov Model. This is due to test positions falling into grid cubes that have not been 'seen' in historical trajectories, and thus there is a lack of training data for the transition probabilities out of states corresponding to these grid cubes. We attempted to solve this using the start state expansion method that we have discussed, and were able to greatly reduce the number of unsuccessful predictions. We believe this idea is worth exploring further, such as through using multiple layers of reference grids, each containing grid cubes of a larger size from the grid cubes of the layer below, such that when a prediction is unsuccessful for the bottom-most layer, another prediction attempt will be triggered using the grid cubes for the layer above.

Thirdly, we designed our Markov Models based on the first-order Markov assumption in order to keep the models computationally simple and fast. This assumption means that the future states (*i.e.* future positions) of the model depends only on the current state (*i.e.* current position), and not on the states that preceded the current state (*i.e.* past positions), and is obviously untrue for aircraft trajectories as they have inertia (*i.e.* a certain speed and direction of motion). We believe that prediction accuracy can be improved further through the use of higher-order or variable order Markov Models for the TP problem.

Lastly, we have used very simple ideas for our baseline models based on conventional approaches, so that through comparisons we can have a better understanding of their characteristics. The median trajectory approach is certainly worth exploring further, such as through finding a median trajectory for each grid cube. We could also investigate the use of efficient index structures such that sub-trajectories of historical trajectories that are similar to the past trajectory (*e.g.* up to 5 minutes before) of the aircraft can be retrieved quickly and be used for trajectory prediction.

References

[1] Avionics International. (2019). Space-based ADS-B: Going Live in the North Atlantic Airspace, [Online]. Available: http://interactive.aviationtoday.com/avionicsmagazine/march-2019/space-based-ads-b-going-live-in-the-north-atlantic-airspace/. 10

[2] S. Ayhan, P. Costas, and H. Samet, "Prescriptive analytics system for long-range aircraft conflict detection and resolution," in *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - SIGSPATIAL '18*, ACM Press, 2018. DOI: 10.1145/3274895.3274947. [Online]. Available: https://doi.org/10.1145/3274895.3274947. 3

[3] S. Ayhan and H. Samet, "DICLERGE," in *Proceedings of the 8th ACM SIGSPATIAL International Workshop on Computational Transportation Science - IWCTS'15*, ACM Press, 2015. DOI: 10.1145/2834882.2834887. [Online]. Available: https://doi.org/10.1145/2834882.2834887. 7, 8

[4] S. Ayhan and H. Samet, "Aircraft Trajectory Prediction Made Easy with Predictive Analytics," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, ACM Press, 2016. DOI: 10.1145/2939672.2939694. [Online]. Available: https://doi.org/10.1145/2939672.2939694. 10, 11, 13, 14, 32

[5] S. Ayhan and H. Samet, "Time series clustering of weather observations in predicting climb phase of aircraft trajectories," in *Proceedings of the 9th ACM SIGSPATIAL International Workshop on Computational Transportation Science - IWCTS '16*, ACM Press, 2016. DOI: 10.1145/3003965.3003968. [Online]. Available: https://doi.org/10.1145/3003965.3003968. 11

[6] L. Basora, J. Morio, and C. Mailhot, "A Trajectory Clustering Framework to Analyse Air Traffic Flows," in *Seventh SESAR Innovation Days*, 2017. 7

[7] S. G. Benjamin, S. S. Weygandt, J. M. Brown, M. Hu, C. R. Alexander, T. G. Smirnova, J. B. Olson, E. P. James, D. C. Dowell, G. A. Grell, H. Lin, S. E. Peckham, T. L. Smith, W. R. Moninger, J. S. Kenyon, and G. S. Manikin, "A North American Hourly Assimilation and Model Forecast Cycle: The Rapid Refresh," *Monthly Weather Review*, vol. 144, no. 4, pp. 1669–1694, Apr. 2016. DOI: 10.1175/mwr-d-15-0242.1. [Online]. Available: https://doi.org/10.1175/mwr-d-15-0242.1. 15, 16, 31, 36

[8] A. Benoit, J. Storey, and S. Swierstra, "Introduction of Accurate Aircraft Trajectory Predictions in Air Traffic Control," on plans and dev for air traffic control syst, Jan. 1975. 9

[9] Federal Aviation Administration (FAA). (2016). Pilot's Handbook of Aeronautical Knowledge, Chapter 11: Aircraft Performance, [Online]. Available: https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/phak/. 15

[10] Federal Aviation Administration (FAA). (2018). NextGen: Modernization of U.S. Airspace, [Online]. Available: https://www.faa.gov/nextgen/. 1

[11] Federal Aviation Administration (FAA). (2018). Performance Based Navigation, [Online]. Available: https://www.faa.gov/nextgen/how_nextgen_works/new_technology/pbn/in_depth/. 6

[12] E. C. Fernández, J. M. Cordero, G. Vouros, N. Pelekis, T. Kravaris, H. Georgiou, G. Fuchs, N. Andrienko, G. Andrienko, E. Casado, D. Scarlatti, P. Costas, and S. Ayhan, "DART: A Machine-Learning Approach to Trajectory Prediction and Demand-Capacity Balancing," in *Seventh SESAR Innovation Days*, 2017. 10

[13] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory Clustering and an Application to Airspace Monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, Dec. 2011. DOI: 10.1109/tits.2011.2160628. [Online]. Available: https://doi.org/10.1109/tits.2011.2160628. 7

[14] GeoTools. (2019). GeoTools: The Open Source JAVA GIS Toolkit, [Online]. Available: http://www.geotools.org/. 32

[15] C. Gong and D. McNally, "A Methodology for Automated Trajectory Prediction Analysis," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, American Institute of Aeronautics and Astronautics, Aug. 2004. DOI: 10.2514/6.2004-4788. [Online]. Available: https://doi.org/10.2514/6.2004-4788. 2, 4, 18

[16] International Air Transport Association (IATA). (2018). IATA Forecast Predicts 8.2 billion Air Travelers in 2037, [Online]. Available: https://www.iata.org/pressroom/pr/Pages/2018-10-24-02.aspx. 1

[17] International Air Transport Association (IATA). (2018). Traveler Numbers Reach New Heights, [Online]. Available: https://www.iata.org/pressroom/pr/Pages/2018-09-06-01.aspx. 1

[18] International Air Transport Association (IATA). (2019). Air Traffic Management, [Online]. Available: https://www.iata.org/whatwedo/ops-infra/air-traffic-management/Pages/index.aspx. 2

[19] International Civil Aviation Organization (ICAO). (2019). Performance-based Navigation: Overview, [Online]. Available: https://www.icao.int/safety/pbn/Pages/Overview.aspx. 6

[20] B. H. Juang and L. R. Rabiner, "Hidden Markov Models for Speech Recognition," *Technometrics*, vol. 33, no. 3, pp. 251–272, Aug. 1991. DOI: 10.1080/00401706.1991.10484833. [Online]. Available: https://doi.org/10.1080/00401706.1991.10484833. 21

[21] D. Jurafsky and J. H. Martin. (2019). Speech and Language Processing (3rd ed.), Draft of 2019 Oct 16, Appendix A: Hidden Markov Models, [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/. 24

[22] D. Jurafsky and J. H. Martin. (2019). Speech and Language Processing (3rd ed.), Draft of 2019 Oct 16, Chapter 8: Part-of-Speech Tagging, [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/. 10, 11, 20

[23] D. Jurafsky and J. H. Martin. (2019). Speech and Language Processing (3rd ed.), Draft of 2019 Oct 16, Chapter 9: Sequence Processing with Recurrent Networks, [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/. 10

[24] J. Kuchar and L. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 179–189, 2000. DOI: 10.1109/6979.898217. [Online]. Available: https://doi.org/10.1109/6979.898217. 10

[25] W. Liu and I. Hwang, "Probabilistic Trajectory Prediction and Conflict Detection for Air Traffic Control," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 6, pp. 1779–1789, Nov. 2011. DOI: 10.2514/1.53645. [Online]. Available: https://doi.org/10.2514/1.53645. 10

[26] C. D. Manning, P. Raghavan, and H. Schutze, *An Introduction to Information Retrieval*. Cambridge University Press, 2008. DOI: 10.1017/cbo9780511809071. [Online]. Available: https://doi.org/10.1017/cbo9780511809071. 23

[27] National Weather Service (National Oceanic and Atmospheric Administration). (2019). Formula for calculation of pressure altitude, [Online]. Available: https://www.weather.gov/media/epz/wxcalc/pressureAltitude.pdf. 15

[28] Nav Canada. (2019). Air Traffic Control, [Online]. Available: http://www.navcanada.ca/EN/about-us/Pages/what-we-do-atc.aspx. 2

[29] Nav Canada. (2019). Space-based Automatic Dependent Surveillance-Broadcast (ADS-B), [Online]. Available: http://www.navcanada.ca/EN/products-and-services/Pages/Space-based-ADS-B.aspx. 10

[30] Network Common Data Form (NetCDF). (2019). Network Common Data Form (NetCDF), [Online]. Available: https://www.unidata.ucar.edu/software/netcdf/. 32

[31] OAG Punctuality League. (2018). On-time performance for airlines and airports and Top 20 busiest routes based on full year data 2017, [Online]. Available: https://www.oag.com/hubfs/Free_Reports/Punctuality_League/2018/PunctualityReport2018.pdf. 4

[32] X. Olive and J. Morio, "Trajectory clustering of air traffic flows around airports," *Aerospace Science and Technology*, vol. 84, pp. 776–781, Jan. 2018. DOI: 10.1016/j.ast.2018.11.031. [Online]. Available: https://doi.org/10.1016/j.ast.2018.11.031. 7, 8

[33] M. Paglione and R. Oaks, "Implementation and Metrics for a Trajectory Prediction Validation Methodology," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, American Institute of Aeronautics and Astronautics, Aug. 2007. DOI: 10.2514/6.2007-6517. [Online]. Available: https://doi.org/10.2514/6.2007-6517. 35

[34] Y. A. Pan, M. A. Nascimento, and J. Sander, "Online Stochastic Prediction of Mid-Flight Aircraft Trajectories," in *Proceedings of the 12th ACM SIGSPATIAL International Workshop on Computational Transportation Science - IWCTS'19*, ACM Press, 2019. DOI: 10.1145/3357000.3366144. [Online]. Available: https://doi.org/10.1145/3357000.3366144. iii, 25, 38

[35] Pew Research Center. (2019). 28% of American adults use mobile and social location-based services, [Online]. Available: https://www.pewinternet.org/2011/09/06/28-of-american-adults-use-mobile-and-social-location-based-services/. 5

[36] Rapid Refresh (RAP). (2019). Rapid Refresh (RAP), [Online]. Available: https://rapidrefresh.noaa.gov. 31, 36

[37] M. Schafer, M. Strohmeier, V. Lenders, I. Martinovic, and M. Wilhelm, "Bringing up OpenSky: A large-scale ADS-B sensor network for research," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, IEEE, Apr. 2014. DOI: 10.1109/ipsn.2014.6846743. [Online]. Available: https://doi.org/10.1109/ipsn.2014.6846743. 31, 36

[38] Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang, "LSTM-based Flight Trajectory Prediction," in *2018 International Joint Conference on Neural Networks (IJCNN)*, IEEE, Jul. 2018. DOI: 10.1109/ijcnn.2018.8489734. [Online]. Available: https://doi.org/10.1109/ijcnn.2018.8489734. 10

[39] Single European Sky ATM Research (SESAR). (2019). Discover SESAR, [Online]. Available: https://www.sesarju.eu/discover-sesar. 2

[40] J. Snyder, "Map Projections – A Working Manual," vol. 1395, Jan. 1987. 15

[41] The OpenSky Network. (2019). The OpenSky Network, [Online]. Available: https://opensky-network.org. 31, 36

[42] US Government. (2019). GPS Applications, [Online]. Available: https://www.gps.gov/applications/. 5

[43] T. T. Warner, *Numerical Weather and Climate Prediction*. Cambridge University Press, 2009. DOI: 10.1017/cbo9780511763243. [Online]. Available: https://doi.org/10.1017/cbo9780511763243. 16

[44] J. L. Yepes, I. Hwang, and M. Rotea, "New Algorithms for Aircraft Intent Inference and Trajectory Prediction," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 2, pp. 370–382, Mar. 2007. DOI: 10.2514/1.26750. [Online]. Available: https://doi.org/10.2514/1.26750. 10

[45] J. Zendulka and M. Pešek, "Mining moving object data," *Open Computer Science*, vol. 2, no. 3, Jan. 2012. DOI: 10.2478/s13537-012-0018-4. [Online]. Available: https://doi.org/10.2478/s13537-012-0018-4. 5

[46] Y. Zheng, "Trajectory Data Mining," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, pp. 1–41, May 2015. DOI: 10.1145/2743025. [Online]. Available: https://doi.org/10.1145/2743025. 6