Probabilistic Prediction of Time To Fly using Quantile Regression Forest

Stanley Förster, Michael Schultz, and Hartmut Fricke
Institute of Logistics and Aviation
TU Dresden
01062 Dresden
[stanley.foerster, michael.schultz, hartmut.fricke]@tu-dresden.de

Abstract—The air traffic is mainly divided into en-route flight segments, arrival and departure segments inside the terminal maneuvering area, and ground operations at the airport. In our contribution we will focus on the prediction of arrival procedures, in particular the time to fly from turn onto final approach course to threshold, which supports utilizing available capacity more efficiently. Most prediction methods developed so far provide a sole point estimate for the time to fly. We see the need to cover the uncertain nature of aircraft movement by the implementation of a probabilistic approach. This becomes very important in cases where the air traffic system is operated at its limits to prevent safety critical incidents, e.g., separation infringements due to very tight separation. Our approach is based on the Quantile Regression Forest technique that is able to provide a measure of uncertainty of the prediction, instead of a single value only. While the data preparation, model training and tuning steps are identical to classic Random Forest methods, in the prediction phase, Quantile Regression Forests provide a quantile function to express the uncertainty of the prediction. After developing the model, we further investigate in the interpretation of the results and provide different ways of deriving a probability function from it. We found, the Skew Normal Distribution provides an affirmative fit to reflect the characteristics of uncertainty in prediction. With this contribution, there becomes a tool available that allows to predict time to fly more sophisticated, depending on the specific needs of the use case.

Keywords—prediction, final approach, time to fly, random forest, quantile regression

I. INTRODUCTION

Airport performance is mainly driven by the efficient use of the limited resources along the aircraft trajectories. In this context, the airport capacity is defined as maximum amount of aircraft movements in a given period of time and provide a defined quality of service (delay). One of the most restricted components of the airport system is the runway (system). Air traffic controllers have to make sure that the runway utilization is maximized dependent on the traffic mix, environmental constraints (e.g., weather or noise), operational modes (arrivals and departures) ensuring safe operations by considering separation standards (see [12] [3] [4]). Capacity efficient operations are becoming more important, since around 1.5 million flights (accounting for 8% of the demand) will not be accommodated in 2040 [5]. Thus, the capacity gap is equivalent to 8 fully-used runways, spread over 17 different states in the European Civil Aviation Conference (ECAC) region, where 21 airports lack capacity. The higher amount of air traffic movements will also lead to increased traffic complexity, especially in high density areas like the terminal maneuvering area (TMA).

This task’s complexity rapidly increases with future separation standards, like RECAT-EU [6], time-based separation [7] or pairwise separation [8], which all lead to enhanced combinatorial complexity due to an increased number of aircraft classes to consider. Therefore, air traffic controllers require for enhanced support tools that help to overcome those challenges and to separate aircraft efficiently.

Giving hints to the air traffic controllers on how to separate aircraft most efficiently requires knowledge about the aircraft’s speed profiles along the extended runway center line and upstream to anticipate the compression effect, so that the support tool’s advisories won’t lead to loss of separation.

Naturally, the air transportation system is both a competitive and collaborative environment, where stakeholders have to optimize their economic benefits considering various restrictions. By giving airport stakeholders access to data from different sources, airports will be able to make more accurate predictions about their operational progress in the next period. In this context, data-driven predictions and machine learning techniques could enhance airport processes [9] to show hidden interdependencies in the complex airport system. The speed and extent, with which data is shared, have massively increased over the last years aiming at a resilient air transportation system [10]. The obligation to equip aircraft with a ADS-B transponder will begin in Europe in June 2020 [11] and in the US almost all air spaces will be reserved only for appropriately equipped aircraft from January 2020 [12]. It is expected that current surveillance systems will be extended with ADS-B and future ground stations will be fully based on this technology, which is significantly cheaper to install and to operate. Due to the simple requirements on the receivers, ADS-B has contributed to the development of online services that display the current air traffic in real time with worldwide receiver networks (depending on the local coverage), such as The OpenSky Network [13] or Flightradar24 (flightradar24.com). This technology also offers a solution for monitoring remote areas and flights over the oceans with space-based ADS-B [14]. Furthermore, the equipment of ground vehicles at aerodromes should enable a more comprehensive monitoring of the traffic situation on the corresponding movement areas at the apron [15].
A. Status quo

In the context of aviation, several methods from the field of artificial intelligence are used to cluster [17] [18] [19] [20], detect anomalies [21] [22] [23] [24] and predict aircraft trajectories [25] [26] [27], develop dynamic airspace designs [28] [29], analyse runway and apron operations [30] [31] [32], determine airport performance including the impact of local weather events [33] [34], and for airport terminal operations (turnaround) [35]. More initiatives to leverage ADS-B open data in order to improve the state of the art are already commonplace, esp. in the field of aircraft modelling [36] [37]. An ADS-B based milestone concept is already developed to efficiently manage even small/medium-sized airports, focusing on arrival flows, runway occupancy, ground trajectories and taxi times [38] (see figure 1).

![ADS-B-based milestone approach for data-driven operational awareness and predictive capabilities](image)

Fig. 1. ADS-B-based milestone approach for data-driven operational awareness and predictive capabilities [38]

In front of this background, using existing runway systems to full capacity becomes a challenging task for all participating stakeholders. One outstanding challenge is the task of establishing and maintaining aircraft separation. Especially the closing effect plays a dominant role in the development of inter-aircraft distance along the final approach. After passing the final approach fix (FAF), aircraft follow individual speed profiles while decelerating to landing speed. Typically, the leading aircraft, which is much closer to the threshold, is slower than the succeeding aircraft that catches up consequently. Hence, aircraft need to be separated much wider on beginning of the final approach leg by adding some buffer on top of required separation, which is consumed by the closing effect. Ideally, the buffer is fully depleted when the preceding aircraft crosses the threshold and distance to the succeeding aircraft equals the required minimum separation.

For modelling the aircraft’s speed profile, we prefer data-driven methods over physics based models, since the latter are always a simplified representation of the real, complex system and neglect contributing factors. Data-driven methods have experienced great attention recently and are expected to provide much better results, especially with today’s availability of big amounts of data in mind.

Time to fly prediction is the focus of research in many different areas. The following studies, where predicting those values for the final approach segment is presented, serve as

enabler for Eurocontrol’s Leading Optimised Runway Delivery (LORD) Tool [39]. LORD is a controller assistance tool that provides an additional indicator on the radar screen. The indicator represents the proposed position for an aircraft behind its predecessor so that when the leading aircraft passes the threshold, the distance between aircraft is as close as possible at the minimum required separation.

In [40] the authors seek to predict the best separation for the following aircraft when the leader is at 4DME (which corresponds to 4 NM ahead of threshold), considering aircraft-individual speed profiles, wind conditions and the resulting closing effect. The authors explicitly derive true air speed (TAS) from the ground speed profile and discretized wind data. The speed profile and time to fly are formulated as functions of aircraft type and wind band. Finally, the authors derive fixed values for additional separation buffer to apply, depending on wind speed and RECAT-EU wake turbulence category, which reflects average behaviour.

The follow up study evaluates different machine learning techniques to improve the prediction of speed profiles and time to fly [41]. Again, the TAS profile is derived from aircraft ground speed and discretized wind data. The authors evaluated six regression techniques, as well as 4 neural network techniques. Predictions are again made uniformly per aircraft-type/wind-band cluster.

The authors of [42] use a similar data set. Again, speed profile and time to fly prediction is based on average values per cluster of aircraft-type and wind-band. However, this time, the authors assume a normal distribution of the actual values and in addition to the mean, track the standard deviation, as well.

B. Focus and structure of the document

In the presented paper, we focus on the area around the final approach, including the upstream part of turn onto ILS (instrument landing system, provide horizontal and vertical flight guidance) course. We develop a methodology to predict time to fly in considering the uncertainty inherent to predictions of the future. This will enable the assessment of possible impacts on safety and capacity.

Controllers required their support tools to propose only actions that will not compromise safety in the first place. Therefore, a prediction method is required that allows to derive information about the uncertainty of the prediction, the prediction interval.

In the following we first present the data our analysis is based on, including a description of the data preparation steps. Section III describes the method for probabilistic prediction of time to fly and provides an overview of the model assessment measures. Afterwards, in section IV the model implementation, concrete training procedure as well as predictions and assessment is provided. Finally the paper ends with a discussion and outlook in section V.
II. DATA PREPARATION

A. Data Description

The flight track data set comprises a six month period of arrivals and departures to/from a major European airport serving over 400,000 flights per year. All flights are recorded within a 40 NM radius around the airport reference point (cf. Arrival Sequencing and Metering Area (ASMA) [43]) and are represented by a meta data record as well as a list of trajectory points. A flight’s metadata contains information about

- Flight Identifier
- Date and Time of Arrival or Departure
- Origin and Destination Airport
- Runway Identifier
- Call Sign
- Aircraft Type

For each flight, the trajectory is represented by a series of records with a four second update interval. Each record contains information about the aircraft’s position (easting, northing, altitude) and velocity (ground speed).

Furthermore, METAR data has been acquired to provide weather information for the full recording period covered by our data set [44]. Each METAR record is composed of a time stamp and the actual report. A report is issued every half an hour at xx:20 and xx:50, respectively. From each report, we extract information about wind (speed and direction), pressure, temperature and the dew point (cf. [45]).

B. Preprocessing

1) Data Selection: In the preprocessing phase, we prepare the data for further analysis and the use in model training and testing phase. This comprises data set selection, merging of different data sets into a common data frame, as well as data imputation, filtering and some additional pre-calculations.

The data used for this study is clustered by aircraft type and landing runway. The proposed methodology is outlined using an exemplary cluster that contains about 5,000 arrivals performed within a six month period. This way, the model training process is more tailored towards specific characteristics and the predictions are more streamlined, instead of having to deal with more diverse data, which is suspected to have a decreasing effect on model accuracy.

For weather data, no pre-selection was made, the full six month period of nearly 9,000 records was included in the base data set. All records that are not required, for instance due to night flying restrictions and the resulting absence of traffic data during that time, is automatically abandoned in the following data merging step.

Finally, we include airport specific information, especially regarding the landing runway. The most important feature here is the location of the threshold, against which distance and time to fly is measured.

2) Data Merging: After all data is obtained and included in the base data set, the flight tracks are merged with airport data. This relates every point of the trajectory with data about the location of the threshold to allow computation of distance to go. Subsequently, for each record of the trajectory, atmospheric data from the METAR data set is attached. Since METAR reports are issued every half an hour, every point of the trajectories that is recorded in between those times is related to the last, most recently available weather data record. We include both, the METAR report time, as well as the track recording time in the feature set. This allows the model to draw conclusion on the time difference during training phase. A schematic view of all the data sets and how they are merged is depicted in figure 2.

Fig. 2. Schematic view on data sets and their relationship.

3) Imputation: During data imputation phase, records with missing data are identified and, if possible, values are inferred. Flight tracks are always fully recorded, therefore, no missing data is found in the data set. Contrarily, several effects lead to incomplete records in the METAR data set.

First, a METAR record may be missing. This is the case for 15 out of the expected 8,928 records, which corresponds to 0.17 %. Another 10 records contain the string ’NIL’, which indicate that no METAR record has been issued for this time. Furthermore, even if available, not every METAR report contains information about all the weather features that are required for the analysis. For instance, an update may concern runway parameters exclusively (e.g. visibility) but not wind parameters or temperature.

For all those cases, data imputation is required to fill missing values in the data set. In the first step we insert ’NIL’ reports where a METAR report is completely missing. This leads to a fully populated data set. However, this does not provide additional information about weather properties like wind or temperature. Therefore, in a second step, all records with missing information about those properties are selected. This comprises 32 records where temperature, dew point and
pressure is missing, as well as 25 records where wind speed is missing. A numerical value for direction of wind is missing in 770 records. This is not only because of missing METAR reports, but there exists a special value ‘VRB’ for low winds (up to 9 knots in this case) from variable directions. In any case, the longest consecutive sequence of missing weather data is of length 7, which equals a four hour period.

For filling up the missing weather data, we use a linear interpolation approach. While this can be done straightforward for features like temperature, dew point, and pressure, wind speed and direction need special handling. Since wind direction is a cyclical feature, a simple linear interpolation is not possible. Instead, wind is treated as a vector that reflects speed (length) and direction, and which is decomposed into its Cartesian components, usually named \( u \) and \( v \). Those components are then interpolated independently of each other and the results are transformed back into two values for speed and direction.

4) Filtering: To reflect the scope of the time-to-fly prediction that was described above, the flight tracks are truncated accordingly. Therefore, a rectangular region around the final approach path is defined that covers the area where aircraft are usually cleared to intercept ILS. This comprises the extended runway centerline within the 40 NM radius of data coverage, and extends orthogonally to this until just before the downwind leg to not include that part of the track. Figure 3 depicts this area schematically.

As soon as an aircraft enters this region, a prediction of its time to fly is possible. Flights, that leave the area after entering it, for instance due to overshooting the LOC intercept, are considered from first entry on, provided they land eventually.

Additional filter criteria are altitude and heading to exclude flights that cross the area at much higher flight level or in a direction that is not appropriate for approach. Furthermore, missed approaches, i.e. flights that did not reach runway threshold, are removed.

5) Feature Transformation: For random forest methods, there is no normalization or standardization of the data required since RF models are based on decision trees, which are invariant to linear transformations because they use indicators like information gain, Gini index, and variance reduction, which are not affected by scaling of the variables. This stays true for quantile regression forest (QRF), even though it is a regression method, because there is no explicit regression coefficient to measure relationship between features and response variable.

Finally, the last step of the data preparation phase is the decomposition of the cyclical features representing wind direction and aircraft heading into their Cartesian components. This step is required, since for cyclical features, the relation between distant values can be much closer than indicated numerically, like for instance headings 005 and 355. To account for this, we split aircraft speed vector and wind speed vector each into two features, one for the sine and one for the cosine part. This corresponds to handling of wind direction during data imputation phase, as described above.

C. Exploratory Data Analysis

After the preprocessing phase, there are 4886 flights available contributing to a total of 452,037 data points. Figure 4 visualizes the tracks that are included in the data set.

For all recorded flight tracks, the time to fly could be determined for every point along the flight path. Figure 5 depicts for all tracks of the data set the time to fly along the flight path. Furthermore, figure 6 visualizes the time to fly distribution using a histogram over all the values in the data set. It highlights the values that are expected at first contact with an aircraft entering the active area. It can be seen that those are not common distributions. The mean time to fly over all the data points is 206.58 seconds, which is depicted by a green line in the diagram.

III. QUANTILE REGRESSION FORESTS

A. Quantile Regression

Predictions made by machine learning (ML) algorithms are never absolute but rather probabilistic. Despite this fact,
most of the ML methods usually provide a point estimate, which reflects only one property of the underlying probability distribution, depending on what the models are training for. Most of the times this is the conditional mean $\mathbb{E}(Y|X = x)$. However, many applications would benefit from more distinct information about the underlying distribution. Sometimes, it is crucial to obtain a prediction interval, rather than a single value. Employing quantile regression methods allow for determining additional features of the probability distribution by predicting several quantiles [46].

For a (conditional) random variable $Y|X = x$ and its cumulative distribution function (CDF) $F(y|X = x)$, the $\tau$th quantile of the random variable is given by equation (2). However, employing quantile regression techniques, the quantiles are what the model provides, and the probability function is what needs to be derived from the outputs. Figure 7 visualizes an exemplary quantile function as constructed from the model’s prediction output for the quantiles from 0 to 100.

$$Q_x(\tau) = F^{-1}(y|X = x)$$

(1)

$$= \inf\{\mu : F(\mu|X = x) \geq \tau\}$$

(2)

Deriving the CDF from the quantile function requires to invert the relationship presented above in equation (1). From the inverted quantile function $Q(\tau)^{-1}$, a suitable CDF $F(y)$ is fitted. The fitting process provides the parameters of the underlying probability distribution, which is the final result of the probabilistic prediction. This distribution can then be used to perform further, uncertainty-based investigations, like a trade-off analysis, etc. Figure 8 visualizes the inverted quantile function from figure 7, as well as a fitted CDF, which represents a Normal Distribution for this example.

![Fig. 6. Distribution of time to fly values. The blue histogram comprises the full data set. The orange one depicts the time to fly at first entry of an aircraft. Solid curves depict the kernel density estimation. The green line depicts the mean over all time to fly values.](image)

![Fig. 7. A quantile function constructed from the responses at 101 collocation points as given by the QRF model.](image)

![Fig. 8. Plot of inverse quantile function (blue) and the fitted cumulative probability function of a Normal Distribution (orange)](image)

Usually, when performing quantile regression, ML models need to be trained towards certain quantiles $\tau$ of interest to provide the required results. Instead of a squared error or absolute error loss function, a specific quantile loss function is employed to determine the performance of the model. In contrast, a QRF model needs to be trained only once and can afterwards provide predictions for arbitrary quantiles, since the random forest method is not based on minimizing a specific loss function [47].

B. Random Forest

Random Forest is an ensemble method based on binary decision trees. A prediction is made based on the outcomes of multiple, distinct trees, hence it’s called a forest. At each level of a tree, the decision, which branch to follow, is made based on the value of certain features. Depending on the objective of the prediction (classification or regression) the trees’ outputs are regarded a vote for a certain class or contribute in the calculation of a numerical value.

In general, a random forest model is trained via a random-ized bootstrapping process [48]. This means, the training data set is sampled $n$ times, depending on the number of trees in the forest. Each sample contains the exact same amount of observations as the original data set. The observations are chosen randomly from the base data set, which leads to the effect of duplicate and missing records. Therefore, this method is called sampling with replacement.

After sampling $n$ individual sets of training data, the training process starts growing one tree per sample. A tree is grown by an iterative splitting of the data sample into two parts based on a decision criterion that gives the reduction in variance to determine the quality of a split [48].

The training process of a QRF model does not differ from training a standard random forest model with the exception of how the values in the leaves of each tree are handled. While in a classic random forest regression model, only the average
of all observations that a leave contains is stored, in QRF all the observations of the training phase are preserved. In the prediction phase, those values are necessary to calculate individual weights for constructing the quantile function as described in detail in [47].

C. Model Quality Assessment

For quantification of model performance and accuracy, there exist several methods [49]. All of those depend on point estimates made by a classic, non-quantile ML method. The calculations of the different measures that are discussed in the following are shown in equations (3) to (5). The root mean squared error (RMSE) is a quite common measure, despite the fact that it is hardly interpretable, it is more suited to compare different models or model parameterizations against each other. In contrast, the mean absolute error (MAE) gives the average deviation of the prediction from the expectation. However, the absolute error integrates a large variety of magnitude both of absolute time to fly values, and prediction errors. For instance, predicting a time to fly of 550 s when the actual value is 500 s is much better than predicting 250 s when the actual value is 200 s. In this example, even though the absolute error is identical, the relative error is 10 % in one case and 25 % in the other. Therefore, we decided to furthermore determine the scale-independent mean absolute percentage error (MAPE) to provide another measure for the quality of the model.

\[
RMSE = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (t_n - \hat{t}_n)^2} \quad (3)
\]
\[
MAE = \frac{1}{N} \sum_{n=1}^{N} |t_n - \hat{t}_n| \quad (4)
\]
\[
MAPE = \frac{1}{N} \sum_{n=1}^{N} \frac{|t_n - \hat{t}_n|}{t_n} \quad (5)
\]

IV. Analysis of Time to Fly Prediction

For implementing the model we use the quantile regression forest implementation of the scikit-forest Python library [50], which is based on the widely used ML framework scikit-learn [51]. The framework provides several methods that help to accomplish standard ML tasks, like train and test data set split, hyper parameter tuning, accuracy measures, etc.

Before training the model, the data set was split into two parts. The training subset comprises 80 % of the available data. The remaining 20 % form the test data set, to assess the model in terms of prediction quality and accuracy.

We started with using only track features, like 3D-position and ground speed of the aircraft. Later, we added weather information from METAR, as described above. The final set of features additionally comprises temporal information, i.e. the time of recording, as well as the time of the latest METAR record. In the following, when presenting results of the prediction and model accuracy measurements, the difference between the feature subsets are outlined.

![Fig. 9. Comparison of expected and predicted time to fly of the model trained using the full set of features. The orange line in the center diagram depicts the y = x diagonal along which prediction and expectation would be identical. The histograms to the top and right of the diagram show the distribution of expected and predicted time to fly values in the test data set, respectively. The blue lines depict the kernel density estimates.](image-url)

Before starting the analysis, it is common practice to create a simple baseline prediction for comparison of the newly developed model. It’s a rather naive approach, where the flight time is predicted simply by returning the average time to fly of the training data set.

Using the training data set, we grew a forest consisting of 500 trees. The QRF implementation provides several hyper parameters, for which it is advised to stay with the default settings. Trials with different settings have shown that hyper parameter tuning gives a marginal improvement in accuracy. Considering the huge amount of training time required, we hence decided to not intensify efforts in hyper parameter tuning at this stage of our research.

The trained QRF model was then used to predict time to fly values based on the test data set. Therefore, the following analysis is based on this subset only, not considering training data. For the model trained on the full set of features, figure 9 visualizes the comparison between expected and predicted time to fly values in the test data set. Furthermore, it depicts the distribution of expected and predicted time to fly values.

In table II the quantitative model error measures – root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) – are summarized for three variants of the model that were trained using the different feature sets, as well as the baseline prediction.

Even though, the model seems to make quite accurate predictions, there is still a mismatch with actual (or estimated) values. Therefore, in the following, we show how the develop-
The default Normal Distribution does not reflect the underlying distribution. However, the middle and right diagrams demonstrate two cases, where the skewness of the distribution becomes obvious. The green line finally visualizes the Skew Normal Distribution fit.

V. CONCLUSION AND OUTLOOK

In this paper, we presented the development of a quantile regression forest model to predict the time to fly of aircraft from turn onto final approach course to threshold. The quality of the model in terms of point estimate accuracy has been investigated regarding several model error measures. Finally, we presented an approach on how to derive a probability distribution to reflect the uncertainty in prediction.

Following the development of this initial mode, the next step will be to extend the model towards implementation in a controller assistant tool. At the current stage, the proposed model considers the current state of the aircraft and weather only. It seems promising to include (partially) the track history as a feature, as well to improve accuracy and precision of the prediction. However, it needs to be investigated whether the QRF approach can be employed for variable length time-series input. Furthermore, we will investigate the possibilities to make use of the proposed approach for predicting flight times to arbitrary points along the nominal flight path, not necessarily to the threshold only. This is required to estimate the closing effect, which means the development of aircraft separation along the final approach leg, where it is necessary to predict time to fly towards a point several miles ahead of threshold, according to separation. Furthermore, the prediction of time to fly and its indication to the air traffic controller is just the first step. With this functionality, the support tool is still a passive tool, where the controller has to decide how to achieve the proposed system state. The next step is to enhance the tool’s capabilities towards an active support tool that additionally generates recommendations for ATC commands, using advanced ML techniques.

ACKNOWLEDGMENT

This work is part of the FUTURE project, which is sponsored by Deutsche Flugsicherung GmbH and the German
Federal Ministry for Economic Affairs and Energy in the course of the national program Luftfahrtforschungsprogramm (LuFo).

REFERENCES