

Learning aircraft operational factors to improve aircraft climb prediction: A large scale multi-airport study

Richard Alligier, David Gianazza

▶ **To cite this version:**

Richard Alligier, David Gianazza. Learning aircraft operational factors to improve aircraft climb prediction: A large scale multi-airport study. Transportation research. Part C, Emerging technologies, 2018, 96, pp.72-95. 10.1016/j.trc.2018.08.012 . hal-01878615

HAL Id: hal-01878615
https://enac.hal.science/hal-01878615

Submitted on 21 Sep 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Learning Aircraft Operational Factors to Improve Aircraft Climb Prediction: A Large Scale Multi-Airport Study

Richard Alligier[a,*], David Gianazza[a]

[a] *ENAC Lab, Toulouse, France*

Abstract

Ground-based aircraft trajectory prediction is a major concern in air traffic control and management. A safe and efficient prediction is a prerequisite to the implementation of new automated tools.

In current operations, trajectory prediction is computed using a physical model. It models the forces acting on the aircraft to predict the successive points of the future trajectory. Using such a model requires knowledge of the aircraft state (mass) and aircraft intent (thrust law, speed intent). Most of this information is not available to ground-based systems.

This paper focuses on the climb phase. We improve the trajectory prediction accuracy by predicting some of the unknown point-mass model parameters. These unknown parameters are the mass and the speed intent. This study relies on ADS-B data coming from The OpenSky Network. It contains the climbing segments of the year 2017 detected by this sensor network. The 11 most frequent aircraft types are studied. The obtained data set contains millions of climbing segments from all over the world. The climbing segments are not filtered according to their altitude. Predictive models returning the missing parameters are learned from this data set, using a Machine Learning method. The trained models are tested on the two last months of the year and compared with a baseline method (BADA used with the mean parameters computed on the first ten months). Compared with this baseline, the Machine Learning approach reduce the RMSE on the altitude by 48 % on average on a 10 minutes horizon prediction. The RMSE on the speed is reduced by 25 % on average. The trajectory prediction is also improved for small climbing segments. Using only information available before the considered aircraft take-off, the Machine Learning method can predict the unknown parameters, reducing the RMSE on the altitude by 25 % on average.

The data set and the Machine Learning code are publicly available.

Keywords: aircraft trajectory prediction, BADA, mass, speed, Machine Learning, Gradient Boosting Machines

*Principal Corresponding Author

Email address: richard.alligier@enac.fr (Richard Alligier)

Introduction

Most applications in Air Traffic Control and Management (ATC/ATM) rely on a ground-based trajectory prediction. It will be even more true with new operational concepts Consortium (2007); Swenson et al. (2006) envisioning trajectory-based operations. An accurate trajectory prediction is required for the new automated tools and algorithms implementing these concepts. Some of the most recent algorithms designed to solve ATM/ATC problems do require to test a large number of "what-if" alternative trajectories and it would be impractical to download them all from the aircraft. As an example, in Prats et al. (2010) an iterative quasi-Newton method is used to find trajectories for departing aircraft, minimizing the noise annoyance. Another example is Chaloulos et al. (2010) where Monte Carlo simulations are used to estimate the risk of conflict between trajectories, in a stochastic environment. Some of the automated tools currently being developped for ATC/ATM can detect and solve conflicts between trajectories, using Genetic Algorithms (Durand et al. (1996)[1]), or Differential Evolution or Particle Swarm Optimization (Vanaret et al. (2012)). In these conflict solving algorithms, each considered maneuver is associated to the trajectory predicted if such a maneuver was issued. If the trajectory prediction is bad, a large safety margin around the predicted trajectories will be taken. As a result, the only remaining conflict free maneuvers might be the one associated to a large cost. With a good trajectory prediction, the safety margin around the predicted trajectories will be smaller. The set of conflict free trajectories will be larger and might contain maneuvers of smaller cost.

Most trajectory predictors rely on a point-mass model to describe the aircraft dynamics. The aircraft is simply modeled as a point with a mass, and the second Newton's law is applied to relate the forces acting on the aircraft to the inertial acceleration of its center of mass. Such a model is formulated as a set of differential algebraic equations that must be integrated over a time interval in order to predict the successive aircraft positions, knowing the aircraft initial state (mass, current thrust setting, position, velocity, bank angle, etc.), atmospheric conditions (wind, temperature), and aircraft intent (thrust profile, speed profile, route). The Eurocontrol Base of Aircraft Data (BADA) project (Mouillet (2017)) implements such a physical model and provides default values for the models parameters.

In current operations, the trajectory is predicted by using the reference mass $mass_{\text{ref}}$ and the reference $(cas_{1\text{ref}}, cas_{2\text{ref}}, M_{\text{ref}})$ values from BADA. The latter values describe the speed profile of a climbing aircraft. The aircraft climbs at constant CAS (Calibrated Airspeed) equals to cas_1 till 10,000 ft, then it accelerates to reach cas_2. It maintains a constant CAS at cas_2 till the transition altitude is reached, then it climbs at a constant Mach M. Although BADA associates one (cas_1, cas_2, M) value to each aircraft type, these values might be different among aircraft of the same type due to different cost-indexes for instance. Figure 1 illustrates how these reference parameters are used to compute the trajectory.

[1]These algorithms are at the root of the strategic deconfliction through speed adjustments developped in the European ERASMUS project (Drogoul et al. (2009)). A more recent application is the SESAR 4.7.2 (*Separation Task in En Route Trajectory-based Environment*) project, where lateral and vertical maneuvers are also used.

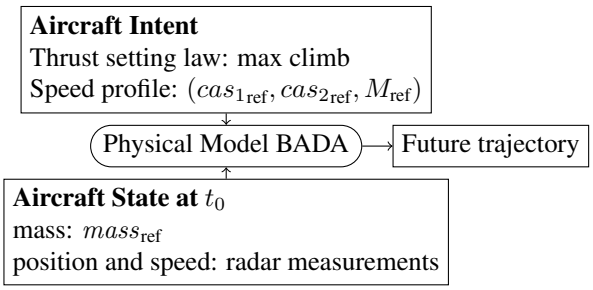


Figure 1: How the reference parameters are used to compute the predicted trajectory

In this paper, we apply Machine Learning methods to predict a mass m and a tuple (cas_1, cas_2, M) that will hopefully provide better trajectory prediction than the default BADA values. The four predictive models h_{mass}, h_{cas_1}, h_{cas_2} and h_M are trained on historical data containing a large number of past flights collected over the first ten of the year 2017. Once built, these models provide a specific prediction for every considered aircraft. For each aircraft, all the information available about this aircraft is embedded in a vector of features x, and the predictive models compute their prediction with x as the input. Figure 2 illustrates how the predicted trajectory is computed using the learned predictive models h_{mass}, h_{cas_1}, h_{cas_2} and h_M. These models are tested on flights collected on the last two months of 2017. All these past flights were collected on ADS-B data by The OpenSky Network. The 11 most frequent aircraft types are studied. The obtained data set contains millions of climbing segments from all over the world. They are tested on trajectory prediction problems with various starting altitude, different climbing segments duration and different prediction time horizons. We have also tested two different sets of variables for the input x: one set containing only variables available before take-off and one set containing all the information available when the aircraft flies.

Previous papers (Alligier et al. (2015b,a)) used a similar approach on Mode-C/Mode-S radar data concerning only two airports. In this paper we use a large ADS-B data set including 1,520 airports. In addition, we consider situations that were previously untested. We also use a more rigorous methodology to determine what would be the performance of our method if it was implemented in an actual operational context. We removed all the identified possible optimistic biases of the performance evaluation.

The rest of the paper is organized as follows: Section 1 describes the context and the approach of this study. Section 2 describes some useful Machine Learning notions that help understanding the methodology applied in this study. Section 3 details the data used in this study. The application of Machine Learning techniques to our operational factors prediction problem is described in section 4, and the results are shown and discussed in section 5, before the conclusion.

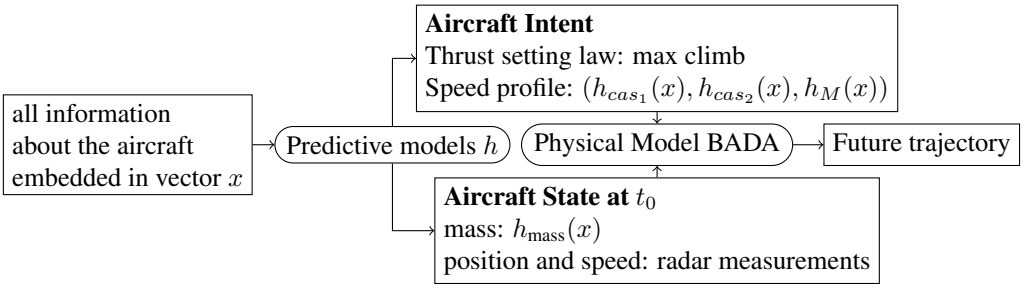


Figure 2: How the predicted trajectory is computed using the learned predictive models h_{mass}, h_{cas_1}, h_{cas_2} and h_M.

1. Context

This section describes previous related works. In this existing context, it also describes the approach followed in this study.

1.1. Literature Review

Some studies (Martin and Mykoniatis (1998); ADA (2009); Coppenbarger (1999)) detail the potential benefits that would be provided by additional or more accurate input data. In other works, the aircraft intent is formalized through the definition of an Aircraft Intent Description Language (López-Leonés et al. (2007); Lopes-Leonés (2007)) that could be used in air-ground data links to transmit some useful data to ground-based applications. All the necessary data required to predict aircraft trajectories might become available to ground systems someday. In the meantime different methods have been designed to obtain these input parameters from the data that is already available today.

Many recent studies (Schultz et al. (2012); Thipphavong et al. (2012); Park and Thipphavong (2013); Alligier et al. (2014); Sun et al. (2016); Uzun and Koyuncu (2017)) used past trajectory points to estimate the aircraft mass using a total energy model such as BADA. All these methods adjust the mass to fit observed values of energy variation. Sun et al. (2017a) proposes a bayesian approach to merge several mass estimates into a refined posterior probability. In Bronsvoort et al. (2015), a mass estimate is extracted from the down-linked Extended Projected Profile (EPP) with the aim to facilitate air-ground trajectory synchronization.

Applying Machine Learning techniques on the trajectory prediction problem is not a new idea. Almost two decades ago, Le Fablec (1999) has applied artificial neural network on this problem. It has also been investigated more recently using different Machine Learning techniques (Tastambekov et al. (2014); Ghasemi Hamed (2014); Hrastovec and Solina (2014, 2016)). With these approaches, the obtained model directly predicts the trajectory or a related metric. It is a black-box hiding what comes from the aircraft performances and what comes from the airline procedures.

Other, less black-box, data-driven approach have been applied to trajectory prediction. Sun et al. (2017b) models a distribution for each aircraft performance parameters.

Each aircraft performance parameter distribution is fitted independently. The prediction provided is not tailored to the considered aircraft. Alligier et al. (2013) fits mean thrust setting profiles using mass estimation methods described in Alligier et al. (2014) and a set of flights. Using Flight Data Recorder (FDR) data and Machine Learning, Chati and Balakrishnan (2017) builds a model that predicts the mass knowing the starting and ending speeds of the takeoff ground roll.

In Alligier et al. (2015a), using Machine Learning, we proposed a model predicting the mass built from Mode-C and Mode-S trajectory data. In Alligier et al. (2015b), Machine Learning is used to predict the (cas_2, M) values of the speed profile. In the two latter publications, the parameters are predicted knowing the airline operator, the departure and arrival airport, the past positions and past speeds. Likewise, these methods were tested on a limited number of airports and for flights with an altitude superior to 18,000 ft.

1.2. Approach Used in this Study

In this publication we apply an approach similar to the one developed in Alligier et al. (2015b,a). However, this study extends the latter articles in many ways and some possible biases are removed.

In this study, Machine Learning techniques are used to predict the operational factors required to compute the predicted trajectory. This study focuses on the climb phase as these operational factors have a huge impact on the trajectory during this phase. These operational factors are the mass m and the speed profile specified by the tuple (cas_1, cas_2, M) as defined in Mouillet (2017). For each considered aircraft type, we build one predictive model for each operational factors. Each predictive model is built using Gradient Boosting Machine (GBM), a supervised Machine Learning technique. As such, it requires a set of examples where the true operational factors are known. We do not have such a data set. However, we can extract the operational factors from the whole climbing trajectory and build a set of examples. The predictive models are obtained by applying GBM to this set of examples. Once the model is trained, new predicted trajectories can be computed using BADA with the predicted operational factors.

Compared with the previous articles (Alligier et al. (2015b,a)), this study uses a more complex methodology. We achieve two goals with this methodology: 1) obtain a performance evaluation that is not optimistically biased; 2) use explanatory variables as they would be obtained in a real-time application. Models are trained on past data (January to October) and tested on future data (November to December) as it would have to be done if such a method was implemented. All the processes involved to compute the explanatory variables use only the past raw points to decide on the current point. It is the case for the process discarding erroneous values and the Kalman filter used to smooth the past trajectory. In the previous publications, we used a smoothing cubic spline applied on the whole trajectory including the future points. One could argue that some information on the future points might leak into the past smoothed points.

Beside the methodology, compared with previous articles, this study tests more airports and more situations. The previous studies using Mode-C/Mode-S data from only two airports whereas this study proves that one model can handle hundreds of

airports at a time and provide good results for each one. It uses a much larger ADS-B data set covering the whole year 2017 and aircraft climbing from airports of all the continents. The previous studies tested the trajectory prediction on a 10 minutes horizon where the current point altitude was superior to 18,000 ft. In this study, there is no requirements on the current altitude. This adds the cas_1 parameter to the speed profile. In addition, we assess the performance of our models – designed to make 10-minutes ahead predictions – on a shorter prediction horizon of 5 minutes. The objective is to check that the quality of the prediction is consistently good for different prediction horizons. We also check the performance of our models when applied to shorter climb segments that were not seen in the training set. Finally, we propose a model that predicts the aircraft operational factors before the aircraft actually takes off, using only information available hours before take-off.

1.3. Data and Machine Learning Code are Publicly Available

Data is the basis of a lot of Air Traffic Management (ATM) research works. However, these data are not easy to obtain (Bourgois and Sfyroeras (2014)). Sharing data could enable comparisons of different approaches on the same basis. Koelle (2017) emphasizes on the benefits of making studies reproducible through the use of open software and open data. We tried to make this study effortlessly reproducible. However, due to the BADA license, we are not able to share the BADA related code. In this study, the data is shared. Except BADA related source code, all the written source code is based on open source software and is also shared.

The data used in this study comes from The OpenSky Network (Schäfer et al. (2014)). It is a participatory sensor network of ADS-B sensors that covers mainly Europe and North-America. It makes the data available to the researchers. These data have been filtered and augmented. The obtained data set, on which this study is based, is available at https://opensky-network.org/datasets/publication-data. Sharing the data set will help others to reproduce this study, design their own algorithm, and compare their approach to ours on the exact same data set. In our opinion, making the data set available will enable scientifically sound comparisons.

The source code is written in Python and OCaml. This code is available at the author's GitHub page https://github.com/richardalligier/trc2018. This code uses open source libraries such as pandas, numpy, matplotlib and LightGBM. With the code shared, everyone can easily reproduce some of the tables and figures in this article. This code is the one used to compute the predicted operational factors. However, the part of the code computing the trajectories is not shared.

The BADA license did not permit us to share the code that computed the predicted trajectories from the predicted operational factors. As a consequence, we could not provide all the code that would make this study effortlessly reproducible. This is quite troublesome because, to the knowledge of the authors, there is no BADA reference implementation that deals with a non standard atmosphere. Such an implementation is not trivial as there are many different ways to benefit from a weather forecast. This is the reason why the subsection 4.5 describes with many details how we have computed the trajectory using the weather forecast.

2. Machine Learning

This section describes some useful Machine Learning notions and techniques. For a more detailed and comprehensive description of these techniques, one can refer to Hastie et al. (2001); Bishop (2006).

As explained in the previous section, we want to predict a variable y, here the adjusted speed profile values cas_1, cas_2 and $Mach$ and the mass value m. We want to predict them from a vector of explanatory variables x, which in our case is the data extracted from the past trajectory points, the weather forecast, and aircraft database containing aircraft type variants, airline operators and routes. This is typically a regression problem. Naively said, we want to learn a function h such that $y = h(x)$ for all (x, y) drawn from the distribution (X, Y). Actually, such a function does not exist, in general. For instance, if two ordered pairs (x, y_1) and (x, y_2) can be drawn with $y_1 \neq y_2$, $h(x)$ cannot be equal to y_1 and y_2 at the same time. In this situation, it is hard to decide which value to give to $h(x)$.

A way to solve this issue is to use a real-valued *loss function* L. This function is defined by the user of function h. The value $L(h(x), y)$ models a cost for the specific use of h when (x, y) is drawn. With this definition, the user wants a function h minimizing the expected loss $R(h)$ defined by equation (1). The value $R(h)$ is also called the *risk*.

$$R(h) = E_{(X,Y)}\left[L\left(h(X), Y\right)\right] \tag{1}$$

However, the main issue when choosing a function h minimizing $R(h)$ is that we do not know the joint distribution (X, Y). We only have a set of examples of this distribution.

2.1. Learning from examples

Let us consider a set of n examples $S = (x_i, y_i)_{1 \leqslant i \leqslant n}$ coming from independent draws of the same joint distribution (X, Y). We can define the *empirical risk* $R_{\text{empirical}}$ by the equation below:

$$R_{\text{empirical}}(h, S) = \frac{1}{|S|} \sum_{(x,y) \in S} L\left(h(x), y\right). \tag{2}$$

Assuming that the values $\left(L(h(x), y)\right)_{(x,y) \in S}$ are independent draws from the same law with a finite mean and variance, we can apply the law of large numbers giving us that $R_{\text{empirical}}(h, S)$ converges to $R(h)$ as $|S|$ approaches $+\infty$.

Thereby, the *empirical risk* is closely related to the *risk*. So, if we have to select h among a set of functions H minimizing $R(h)$, using a set of examples S, we select h minimizing $R_{\text{empirical}}(h, S)$. This principle is called the *principle of empirical risk minimization*.

Unfortunately, choosing h minimizing $R_{\text{empirical}}(h, S)$ will not always give us h minimizing $R(h)$. Actually, it depends on the "size"[2] of H and the number of examples

[2]The "size" of H refers here to the complexity of the candidate models contained in H, and hence to

$|S|$ (Vapnik and Chervonenkis (1991); Vapnik (1995)). The smaller H and the larger $|S|$ are, the more the *principle of empirical risk minimization* is relevant. When these conditions are not satisfied, the selected h will probably have a high $R(h)$ despite a low $R_{\text{empirical}}(h, S)$. In this case, the function h is *over-fitting* the examples S.

These general considerations above have practical consequences on the use of Machine Learning. Let us denote h_S the function in H minimizing $R_{\text{empirical}}(., S)$. The *risk* using h_S is given by $R(h_S)$. We use the *principle of empirical risk minimization*. As stated above, some conditions are required for this principle to be relevant. Concerning the size of the set of examples S: the larger, the better. Concerning the size of H, there is a trade-off: the larger H is, the smaller $\underset{h \in H}{min}\ R(h)$ is. However, the larger H is, the larger the gap between $R(h_S)$ and $\underset{h \in H}{min}\ R(h)$ becomes. This is often referred to as the *bias-variance trade-off*.

2.2. Accuracy Estimation

In this subsection, we want to estimate the accuracy obtained using a Machine Learning algorithm $\mathcal{A}$. Let us denote $\mathcal{A}[S]$ the prediction model found by algorithm $\mathcal{A}$ when minimizing $R_{\text{empirical}}(., S)$[3], considering a set of examples S.

The *empirical risk* $R_{\text{empirical}}(\mathcal{A}[S], S)$ is not a suitable estimation of $R(\mathcal{A}[S])$: the law of large numbers does not apply here because the predictor $\mathcal{A}[S]$ is neither fixed nor independent from the set of examples S.

One way to handle this is to split the set of examples S into two independent subsets: a *training set* S_{Train} and another set S_{Other} that is used to estimate the *risk* of $\mathcal{A}[S_{Train}]$, the model learned on the training set S_{Train}. For that purpose, one can compute the holdout error $Err_{\text{hold-out}}$ as defined by the equation below:

$$Err_{\text{hold-out}}(\mathcal{A}, S_{Train}, S_{Other}) = R_{\text{empirical}}(\mathcal{A}[S_{Train}], S_{Other}). \tag{3}$$

The accuracy estimation has basically two purposes: first, model selection in which we select the "best" model using accuracy measurements and second, model assessment in which we estimate the accuracy of the selected model. For model selection, the set S_{Val} in $Err_{\text{hold-out}}(\mathcal{A}, S_{Train}, S_{Val})$ is called *validation set*. For model assessment, the set S_{test} in $Err_{\text{hold-out}}(\mathcal{A}, S_{Train}, S_{Test})$ is called this set is called *test set*.

2.2.1. Model Selection: Hyperparameter Tuning

Some learning algorithms have hyperparameters. These hyperparameters λ are the parameters of the learning algorithm $\mathcal{A}_\lambda$. These parameters cannot be adjusted using the *empirical risk* because most of the hyperparameters are directly or indirectly related to the size of H. Thus, if the *empirical risk* was used, the selected hyperparameters would always be the ones associated to the largest H.

---

their capability to adjust to complex data. As an example, if H is a set of polynomial functions, we can define the "size" of H as the highest degree of the functions contained in H. In classification problems, the "size" of H can be formalized as the Vapnik-Chervonenkis dimension.

[3]Actually, depending on the nature of the minimization problem and chosen algorithm, this predictor $\mathcal{A}[S]$ might not be the global optimum for $R_{\text{empirical}}(., S)$, especially if the underlying optimization problem is handled by local optimization methods.

These hyperparameters allow us to control the size of H in order to deal with the *bias-variance trade-off*. These hyperparameters can be tuned using the holdout method on a *validation set* for accuracy estimation. In order to find λ minimizing the accuracy estimation, we used a grid search which consists in an exhaustive search on a predefined set of hyperparameters. The Algorithm 1 is a learning algorithm without any hyperparameters. In this algorithm, a *training set* S_{Train} and a *validation set* S_{Val} are considered. This algorithm computes the best hyperparameter λ^*.

Algorithm 1 Hyperparameters tuning for an algorithm $\mathcal{A}_\lambda$ and a *training set* S_{Train} and a *validation set* S_{Val}.

function TUNEGRID($\mathcal{A}_\lambda$,$grid$)[S_{Train}, S_{Val}]
  $\lambda^* \leftarrow \underset{\lambda \in grid}{argmin}\ Err_{\text{hold-out}}(\mathcal{A}_\lambda, S_{Train}, S_{Val})$
  return λ^*
end function

2.2.2. Model Assessment: Accuracy When Used on Unseen Data

In the previous subsection, we have seen how to select the best hyperparameter λ^* using S_{Train} and S_{Val}. Using λ^*, we can train our model on $S_{Train} \cup S_{Val}$. We can compute how well it will perform on unseen data by computing $Err_{\text{hold-out}}(\mathcal{A}_{\lambda^*}, S_{Train} \cup S_{Val}, S_{Test})$. Please note that, the three sets used must be disjoint. Figure 3 summarizes the whole process from model selection to model assessment. On this Figure, the model selection is done on steps 1 and 2 whereas the model assessment is done on step 3.

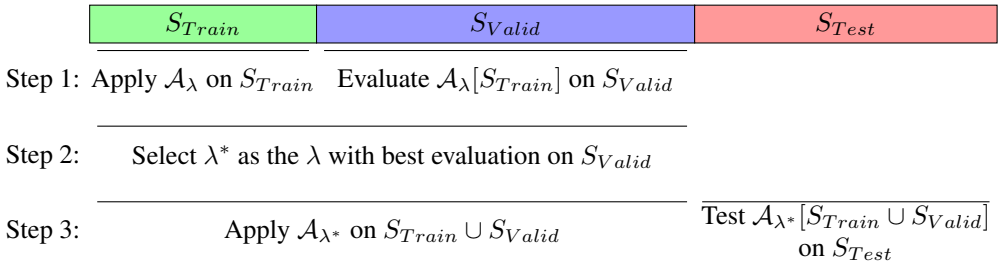


Figure 3: A typical Machine Learning workflow involving a *training set*, a *validation set* and a *test set*.

2.3. Gradient Boosting

The stochastic gradient boosting tree algorithm was introduced in Friedman (2002). It applies functional gradient descent (Friedman (2000) to regression trees Breiman et al. (1984).

The functional gradient descent is a *boosting* technique where the model h is iteratively improved. At each iteration we consider the gradient of the loss $g_i = \frac{\partial L(\hat{y}, y_i)}{\partial \hat{y}}(h(x_i), y_i)$. A Machine Learning algorithm is applied at each iteration to a modified Machine Learning problem where the set of examples is $(x_i, g_i)_{1 \leqslant i \leqslant n}$.

Then the model g obtained is used to update the model h: the updated model is $h^{+1}(x) = h(x) - \rho g(x)$, where ρ is a constant minimizing the empirical risk. In the next iteration we consider h^{+1} instead of h.

In the Gradient Tree Boosting, the Machine Learning algorithm used in the functional gradient descent is a regression tree algorithm Breiman et al. (1984). The model obtained by this algorithm is a binary tree representing a binary recursive partition of the input space. At each node, the input space is split in two according to a condition $x_j \leqslant s$. The J leaves describe a partition $(R_j)_{1 \leqslant j \leqslant J}$ of the input space. Each region R_j is associated to a constant γ_j which is returned by the model when x falls into R_j.

Regression trees have some advantages. They are insensitive to input monotonic transformations. Using x_j, $\log(x_j)$ or $\exp(x_j)$ leads to the same model. As a consequence, this algorithm is robust to outliers. It can easily handle categorical variables and missing values. However, regression trees, used alone, are known to have a poor performance in prediction. Nonetheless, used in combination with a functional gradient descent like the one used in gradient boosting, this latter drawback is very limited. This is the reason why we are using a stochastic gradient boosting tree algorithm.

3. Data Used in this Study

The trajectory data used in this study are from The OpenSky Network (Schäfer et al. (2014)). The OpenSky Network is a participatory sensor network of ADS-B sensors that covers mainly Europe and North-America. The data used in this study covers the year 2017. The augmented and sampled climbing segments used in this study are available at https://opensky-network.org/datasets/publication-data. For this study we have considered the 10 most frequent aircraft types according to Centre (2017). These 10 aircraft types cover 63 % of the European air traffic according to Centre (2017). Actually, in a recent ICAO 8643 document update, the E190 aircraft type designator has been split into two types namely E190 and E195. This leads us to consider 11 aircraft types.

3.1. From Raw Trajectory Points to Clean Sampled Climbing Segments

From The OpenSky Network, we have downloaded all the raw trajectory points of the year 2017 with a vertical rate superior or equal to 256 feet/min. These raw points are processed to obtain clean sampled climbing segments. The process used is summarized in Figure 4.

3.1.1. Discarding Erroneous Values

Some raw trajectory points contain values that are obviously erroneous[4]. In order to discard erroneous values, we consider the time-ordered sequence of values. The first value is assumed to be correct. Then, the next value is considered to be correct if it is close enough to the previous correct value: it must be consistent with the minimum and maximum rate of variation of the variable. We repeat this process till the last

[4]For instance, a ground velocity of 2,650 m/s for a B738 or an altitude of 38,130 m for a A320.

value of the sequence is reached. This process is described with more details by the Algorithm 2. The parameters of this algorithm are chosen to discard only erroneous values[5].

Algorithm 2 Discards the erroneous values.

Input:

$vmin$ and $vmax$ are the minimum and maximum acceptable values
$dvmin$ and $dvmax$ are the minimum and maximum rates of variation
$(t_i)_{i\in[\![1;n]\!]}$ and $(v_i)_{i\in[\![1;n]\!]}$ are vectors; the value v_i was observed at the time t_i

Output: Vector containing the indexes of the values kept

function DISCARDERRONEOUS($vmin$,$vmax$,$dvmin$,$dvmax$,$(t_i)_{i\in[\![1;n]\!]}$,$(v_i)_{i\in[\![1;n]\!]}$)
  $i_{\text{correct}} \leftarrow 1$
  $seq_{\text{correct}} \leftarrow [i_{\text{correct}}]$
  for $i = 2$ to n **do**
    $\Delta_t \leftarrow t_i - t_{i_{\text{correct}}}$
    if $vmin \leqslant v_i \leqslant vmax$ and $dvmin \times \Delta_t \leqslant v_i - v_{i_{\text{correct}}} \leqslant dvmax \times \Delta_t$ **then**
      $i_{\text{correct}} \leftarrow i$
      $seq_{\text{correct}} \leftarrow concatenate(seq_{\text{correct}}, [i_{\text{correct}}])$
    end if
  end for
  return seq_{correct}
end function

3.1.2. Splitting Data into Climbing Segments

The points associated to the same aircraft are identified using the ICAO 24-bit values. As only positions with a positive vertical rate were downloaded, we only have points in climb phase. However, the sequence of points associated to one aircraft can contain several climbing segments. It may even contain different flights of the considered aircraft. We have to split this sequence of points into climbing segments. Moreover, we have decided that each climbing segment must contain at least one raw point every 30 seconds. The purpose of this requirement is to ensure the quality of the climbing segments we handle in this study.

Consequently, we have split the sequence of points into sub-sequences of points with no time hole superior to 30 seconds. These sub-sequences will be our climbing segments. Please note that two climbing segments can come from the same continuous climb if somehow no position update has been received within 30 seconds during this climb. Conversely, two different continuous climbs will give us two different climbing segments as the two continuous climbs are most likely 30 seconds apart.

[5]For instance, for the altitude we have chose $vmin = -500$ m, $vmax = 17{,}000$ m, $dvmin = -50$ m/s and $dvmax = 50$ m/s.

3.1.3. Smoothing the Climbing Segments

The algorithm described in sub-section 4.2.1 uses a physical model and the derivatives of the airspeed and altitude to estimate the mass. The computation of these derivatives can be more reliable if a smoothing procedure is applied beforehand. In our study, as described in section 4.1, the climbing segments are split in two parts, the past points and the future points. The mass estimation algorithm is used in two different contexts in our study.

In the first context, the mass estimation algorithm is used to compute the variable to be learned by the Machine Learning algorithms. To do this, we apply the mass estimation algorithm on the future points. In this context, we know all the points of the climbing segments. As a consequence, we can use a smoothing cubic spline on all the points. One advantage of a cubic spline is that the estimated derivatives are much more reliable than a simple difference on the raw points. This technique was also used in Alligier et al. (2015a,b).

A second context is the mass prediction. In this context, we apply the mass estimation algorithm on the past points. The mass obtained can then be used to predict the trajectory. With a smoothing cubic spline, if we consider all the points of the climbing segment, the future points will have an impact on how the past points will be smoothed. One could argue that some information on the future points might leak into the past smoothed points. As a consequence we used a Kalman filter (Brown et al. (1992)) to smooth the past trajectory points only. The derivatives are then estimated with a simple difference on the smoothed points. As opposed to the smoothing cubic spline, with the Kalman filter, the value of one point depends only on the points before the considered point. Thus, the smoothed points used are the same we would obtain in a real-time application implementing the same Kalman filter. Please note that the Algorithm 2 used to discard the erroneous values also uses only previous points to decide on the current point.

We also keep raw points in our data set, these raw points will be the reference points. We will compare the predicted trajectories to these non-smoothed points.

Figure 4 describes the different smoothed trajectories obtained through these processes.

3.1.4. Sampling the Climbing Segments

For each segment, we typically have one point every 2-3 seconds. It is more convenient to have climbing segments with points evenly spaced through time. We chose to have one point every 15 seconds. This value seemed to be a good compromise between the number of points used to describe the climbing segment and the precision of this description.

We consider a climbing segment containing n raw points where $(t_i)_{i\in[\![1;n]\!]}$ is the date of the points and $(v_i)_{i\in[\![1;n]\!]}$ is the variable to interpolate. In order to sample the trajectory, we consider the sampling dates $ts_i = t_1 + i \times \Delta_t$, with $\Delta_t = 15$ s. In order to interpolate the variable at ts_i, we search for j such that $t_j \leqslant ts_i < t_{j+1}$. Then we only have to linearly interpolate the values v_j and v_{j+1}.

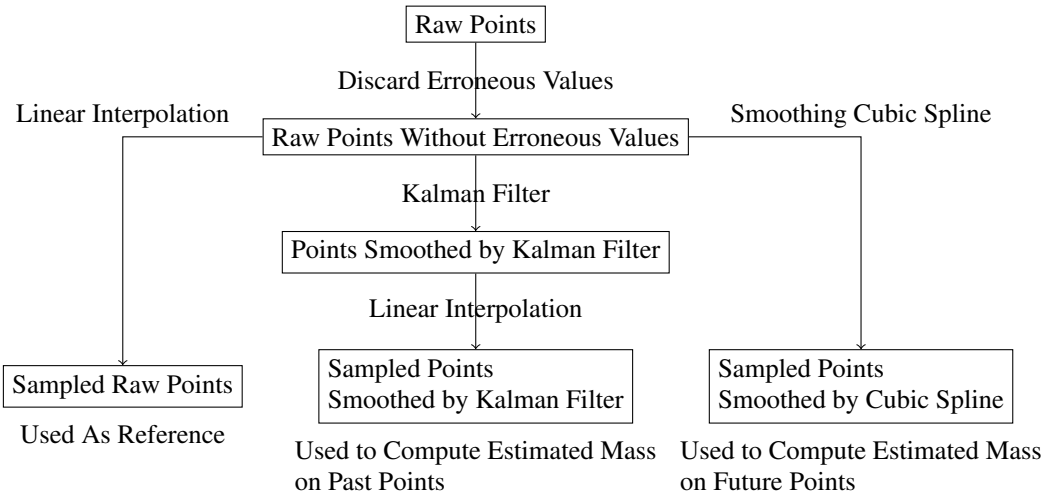


Figure 4: Summarizes the processes used to obtain the clean sampled climbing segments.

3.2. Adding Relevant Information to our Data

Adding information to our data is a mandatory step as the climbing segments do not contain the aircraft type nor the weather for example. These two information are very important in trajectory prediction.

3.2.1. Aircraft Type, Aircraft Variant and Airline Operator

The aircraft type was identified using the ICAO 24-bit address in our segments. Using this address, the aircraft type was retrieved from several databases. For this purpose, an aircraft database was built using VirtualRadarServer[6] and its database writer plugin. If this database did not contain the ICAO 24-bit we were looking for, then we searched it in the World Aircraft Database Sun (2017). In addition to the aircraft type, this database contains the aircraft variant and airline operator. When available, we have also added this information to our segments.

In a recent ICAO 8643 document update, the E190 aircraft type designator has been split into two types namely E190 and E195. Using the aircraft variant we have corrected the aircraft type in our database. This is useful as BADA 3.14 provides a different model for these two aircraft types.

3.2.2. Weather

The Global Forecast System (GFS) was used to add the weather to our segments. More precisely, we have used the forecast files, not the analysis files, with a 1-degree grid. We have one weather grid every 3 hours. A sequential linear interpolation was applied to retrieve the weather at a specific position and time. The temperature, the north

[6]http://www.virtualradarserver.co.uk/

and east wind components were added to each aircraft positions. These temperature and wind values are added because of their impact on the climbing phase.

3.2.3. Departure and Arrival airports

Using the callsign in our segments and the route database from FlightAirMap[7], the departure and arrival airports were identified. We have added the trip distance to our segments by computing the distance between the two airports. This trip distance will provide information on the fuel load and hence the mass of the aircraft which affects the climb.

3.3. Keeping Climbing Segments with a Sufficient Length

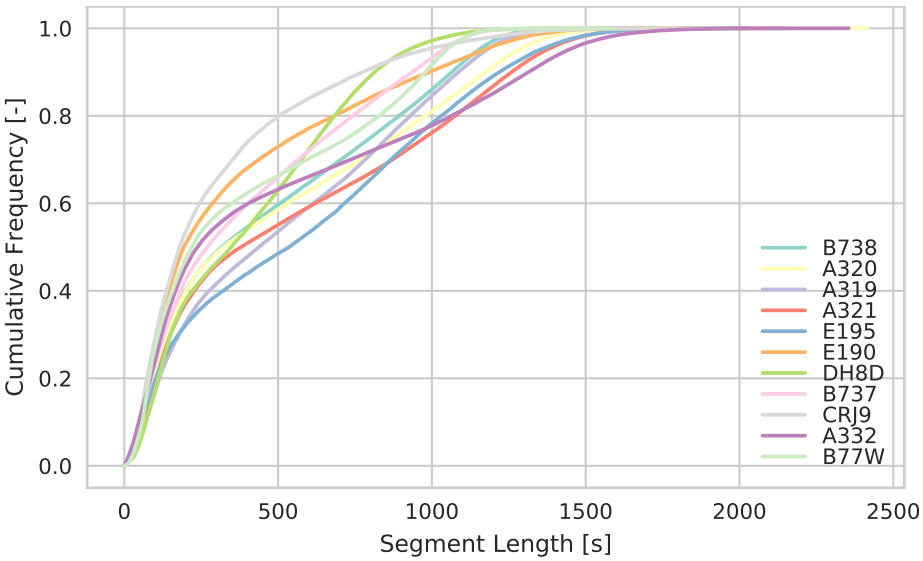


Figure 5: Cumulative distribution of the length of the climbing segments for November and December.

Figure 5 plots the cumulative distribution of the length of the climbing segments for the months of November and December. We can see on this Figure that the segment length distribution is different from one aircraft type to another. In this publication, we build the models from the ten first months of the year, and we test the trained models on the two remaining months.

From January to October, the segments with a duration superior to 750 seconds are kept. By doing so, we discard 60 % to 90 % of the climbing segments according to Figure 5. The mass and speed profile we learn are extracted from each climbing segment. The more points we use to extract these values the more these estimated values are reliable. However, if more points are required then more small climbing segments will be discarded. It leads to smaller training/validation sets. There is a

[7] https://data.flightairmap.com/

trade-off between the size of the training/validation sets and the reliability of the values you use to learn your models. This trade-off is the reason not to include all length of segments in the training set. In our study, we chose a 750 s threshold. This was the obvious value as we were originally interested in predicting a 600 s horizon with a 150 s past. Maybe it is not the optimal value regarding the mentioned trade-off, we did not perform tests on this matter.

For November and December, the climbing segment with a duration superior to 300 seconds are kept. This will allow us to test if models learned on longer climbing segments can be useful to predict trajectory for shorter climbing segments. Even with this 300 seconds threshold, we discard 35 % to 70 % of the climbing segments according to Figure 5.

3.4. Statistics on the Climbing Segments Used in Machine Learning

The number of climbing segments obtained for each aircraft type for the last two months are presented in Table 1. They are sorted from the more frequent to the less frequent aircraft type according to Centre (2017). We can see that this order is not consistent with the number of segments we found for each aircraft type. Actually, one flight can be counted several times if it performs several continuous climbs. This relative order can also be explained by the fact that our traffic contains also the North-American traffic.

Table 1: This table summarizes the number of climbing segments with a duration superior to 300 seconds from November to December. All these climbing segments will be used for testing the models.

model	B738	A320	A319	A321	E195
count	238,198	220,432	87,688	102,745	12,031

model	E190	DH8D	B737	CRJ9	A332	B77W
count	5,695	5,501	26,628	6,168	19,349	23,933

The number of climbing segments obtained for each aircraft for the first ten months are presented in the Table 2. The relative order is also not the one described in Centre (2017). These ten months only includes segments that last more than 750 seconds, as the others were discarded. Each flight is less likely to contain a large number of such long climbing segments. However, the fact that we filtered the climbing segments according to their length might bias the distribution as can be seen in Figure 5. As these segments are sufficiently long, all these segments will be used to train our models.

Figure 6 plots the distribution of the segments' points as a function of the altitude. We can see that the turboprop DH8D has a distribution different from the jets. The jets share a very similar distribution with a peak after 10,000 ft. This peak can be explained by the fact that each points are separated by the same time interval and the fact that the aircraft reduce temporarily their rate of climb after 10,000 ft in order to accelerate.

Figure 7 plots the climbing segments on a world map. In order to produce this figure, 331 millions aircraft positions were aggregated. With this Figure, we can see that the five continents contains climbing segments. However, most of them are located in Europe and North-America. Africa contains the fewest climbing segments.

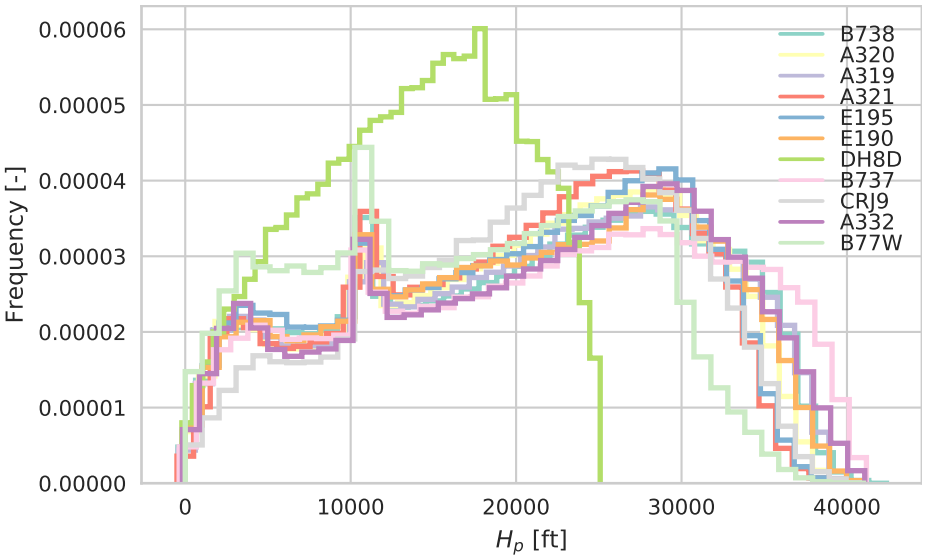


Figure 6: Distribution of the points' altitude of the climbing segments.

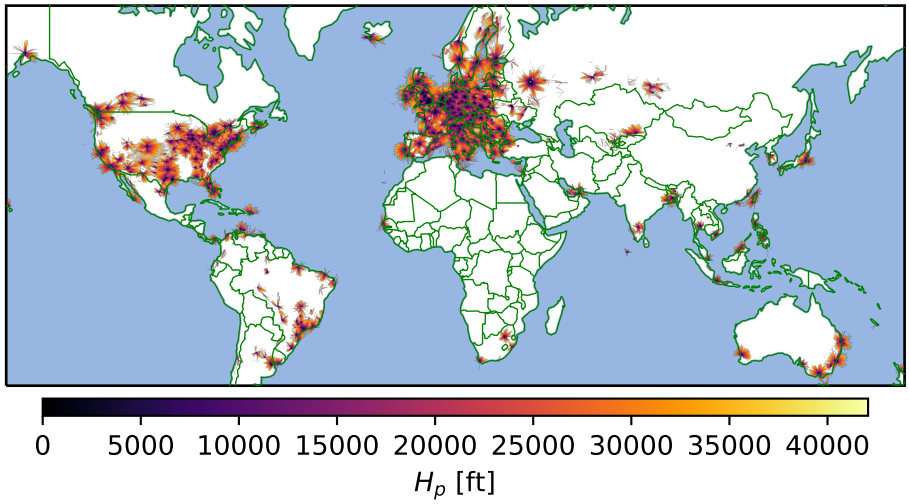


Figure 7: Climbing segments plotted on a world map. The mean altitude can be read from the color.

Table 2: This table summarizes the number of climbing segments with a duration superior to 750 seconds from January to October. All these climbing segments will be used for training our models.

model	B738	A320	A319	A321	E195
count	1,100,156	1,115,773	474,213	492,381	56,642

model	E190	DH8D	B737	CRJ9	A332	B77W
count	33,664	21,896	121,390	20,892	89,881	99,043

Table 3 shows the 10 most frequent departure airports for our segments. We can see that the departure airport is not known for $1.77\ \%$ of the climbing segments. Among these airports, only KLAX is not an European airports. We have 1,522 different departure airports. However, the 37 most frequent departure airports account for $50.5\ \%$ of the climbing segments.

Table 3: This table summarizes the distribution of the departure airports of the climbing segments for the whole year 2017.

ICAO airport code	airport name	frequency
EDDF	Frankfurt Airport	2.38
EGLL	Heathrow Airport	2.27
EDDM	Munich Airport	2.21
EHAM	Amsterdam Airport Schiphol	2.19
LFPG	Paris Charles de Gaulle Airport	1.93
KLAX	Los Angeles International Airport	1.88
EGKK	Gatwick Airport	1.87
-	airport unavailable	1.77
LTBA	Istanbul Atatürk Airport	1.68
LEMD	Adolfo Suárez Madrid–Barajas Airport	1.56

4. Applying Machine Learning to Our Prediction Problem

This section describes how the climbing segments are processed to obtain sets of examples that are suited for Machine Learning. As we want to predict the future trajectory with a certain horizon, we do not use the entire climbing segment to obtain one example. Actually, we can build several examples using different trajectory samples of the same climbing segments.

This section also describes how gradient boosting is applied and how the predicted trajectory is computed using the operational factors and a temperature profile.

4.1. Extracting Trajectory Samples from One Climbing Segment

In this study, knowing the current position and $p = 9$ consecutive past points, we want to predict the future q points. In this context, a trajectory sample is defined by the

current position, the p past points and the q future points. The trajectory samples will be used to train and evaluate our predictive models. These trajectory samples are built from the climbing segments. Actually, we build several trajectory samples from one climbing segment.

Considering one climbing segment with n points, a trajectory sample is built from $p + q + 1$ consecutive points chosen among the n segment points. Hence, from one segment we build $n - p - q$ trajectory samples. Figure 8 illustrates two different trajectory samples (with $q = 40$) extracted from the same climbing segment.

The climbing segments in the first ten months are used to train our models. For these months we have used $q = 40$. Concerning the last two months, we actually produced three different sets: one with $q = 40$, one with $q = 20$ and the last one with $q = 10$. These three different sets will be useful to test how our predictive models perform on different prediction time horizons.

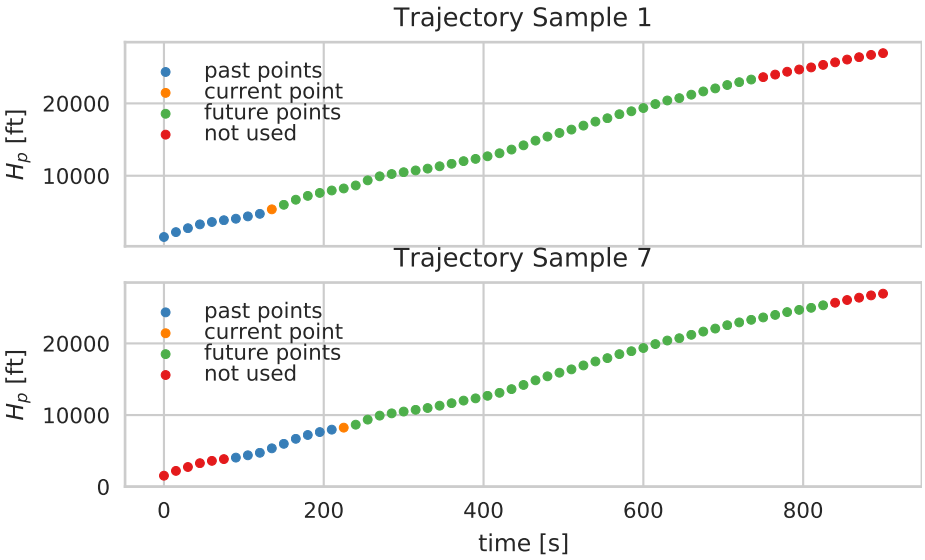


Figure 8: With $q = 40$, two different trajectory samples extracted from the same climbing segments.

4.2. Building One Example from One Trajectory Sample

This subsection describes how the four operational factors have been added to the trajectory samples, allowing us to build the *training set*. In this study, we have to predict four operational factors: the mass m, and three variables (cas_1, cas_2, M) that describe the speed profile used by the aircraft. These factors are predicted using supervised Machine Learning. This technique requires a set of examples, a *training set*, where each example contains the explanatory variables and the variable to be learned. The explanatory variables contain knowledge from the current point and the 9 past points. Hence, each example is derived from one trajectory sample. However, concerning the variable to be learned, the trajectory sample does not contains a mass variable nor

the (cas_1, cas_2, M) speed variables. This subsection describes how these required variables have been extracted from the trajectory points.

4.2.1. Adding the Mass

For each trajectory sample, the mass is estimated using the $q = 40$ future points. The method used to extract the mass from these future points is the one described in Alligier et al. (2014). The mass is estimated by minimizing the difference between the modeled power and the observed energy variation. These two quantities are equals according to the Newton's second law giving us Equation 4 where m is the mass, V_a is the True AirSpeed (TAS), H_p is the altitude, Thr is the thrust, D is the drag, T is the temperature and T_{ISA} is the temperature in ISA standard atmosphere.

$$\frac{\mathit{Thr} - D}{m} V_a = g_0 \frac{T}{T_{ISA}} \frac{dH_p}{dt} + V_a \frac{dV_a}{dt} \tag{4}$$

We have to compute the derivatives $\frac{dV_a}{dt}$ and $\frac{dH_p}{dt}$. Cubic splines are used to smooth the future points, they were also used to compute these derivatives. Let us note E_v the energy variation, t the date. At point i, we have:

$$\frac{\mathit{Power}_i}{m_i} = \frac{\mathit{Thr}_i - D_i}{m_i} V_{ai} \tag{5a}$$

$$\frac{E_{vi}}{m_i} \simeq g_0 \frac{T_i}{T_{ISAi}} \left(\frac{dH_p}{dt}\right)_i + V_{ai} \left(\frac{dV_a}{dt}\right)_i \tag{5b}$$

We can compute $\frac{E_{vi}}{m_i}$ using the past points and a weather model. Using a model of forces such as BADA, Power_i can be computed as a function of H_{p_i}, V_{ai}, T_i and m_i. According to Newton's laws, these two quantities are equal. The mass to be learned will be the one minimizing the sum defined by equation (6) where $i = 10$ is the index of the current point in the trajectory sample and ff is the BADA function modelling the fuel consumption. As described in Alligier et al. (2014), minimizing such a sum can be done efficiently by finding the roots of a fourth degree polynomial.

$$m_{10} = \underset{m_{10}}{\mathit{argmin}} \sum_{i=10}^{10+q} \left(\frac{\mathit{Power}_i(H_{p_i}, V_{ai}, T_i, m_i)}{m_i} - \frac{E_{vi}}{m_i} \right)^2 \tag{6a}$$

$$\text{with } m_{i+1} = m_i - \mathit{ff}\left(V_a, H_p, T\right)\left(t_{i+1} - t_i\right) \tag{6b}$$

We have also applied this technique to estimate the mass on the past points. This mass estimated on past the points is added to the explanatory variable.

4.2.2. Adding the Speed Profile

The speed profile is modeled in BADA with three parameters cas_1, cas_2 and M. This speed profile specify the TAS for a given altitude H_p and a temperature T. The aircraft climbs at a constant Calibrated AirSpeed (CAS) equal to cas_1 from 3,000 ft to 10,000 ft. Then, it accelerates till it reaches cas_2. It climbs at a constant CAS cas_2 till

it reaches the Mach number M. Then it climbs at a constant Mach number. The TAS profile obtained in the end is given by the equation below where f is a function given by Mouillet (2017), T is the temperature, R and κ are physical constants.

$$V_a(cas_1, cas_2, M; H_p, T) = \begin{cases} f(cas_1, H_p, T) & \text{if 6,000 ft} \leqslant H_p < \text{10,000 ft} \\ f(cas_2, H_p, T) & \text{if 10,000 ft} \leqslant H_p \leqslant H_{p_{\text{trans}}}(cas_2, M) \\ M\sqrt{\kappa RT} & \text{if } H_p > H_{p_{\text{trans}}}(cas_2, M) \end{cases} \tag{7}$$

We want to extract cas_1, cas_2 and M from the points in the trajectory sample. Considering Equation (7), we can see that extracting a speed profile requires points from low altitude to high altitude. As a consequence, to extract the speed profile, we consider all the points in the climbing segment, not only the points in the trajectory sample. Hence, all the trajectory samples coming from the same climbing segment will have the same common (cas_1, cas_2, M) given by the equation (8).

$$(cas_1, cas_2, M) = \underset{(cas_1, cas_2, M)}{argmin} \sum_{i=1}^{n} \left(V_a(cas_1, cas_2, M; H_{p_i}, T_i) - V_{ai}\right)^2 \tag{8}$$

As Equation (7) omits the acceleration from cas_1 to cas_2, this optimization problem can be split in two independent parts:

$$cas_1 = \underset{cas_1}{argmin} \sum_{\text{3,000 ft} \leqslant H_{p_i} < \text{10,000 ft}} \left(f(cas_1, H_{p_i}, T_i) - V_{ai}\right)^2 \tag{9a}$$

$$(cas_2, M) = \underset{(cas_2, M)}{argmin} \sum_{\text{10,000 ft} \leqslant H_{p_i} \leqslant H_{p_{\text{trans}}}(cas_2, M)} \left(f(cas_2, H_{p_i}, T_i) - V_{ai}\right)^2 + \sum_{H_{p_{\text{trans}}}(cas_2, M) < H_{p_i}} \left(M\sqrt{\kappa RT_i} - V_{ai}\right)^2 \tag{9b}$$

These two problems can be solved separately using the efficient technique described in Alligier et al. (2015b). Considering n points, this technique consists in splitting the problem into $2n$ sub-problems and solve each sub-problem through a dichotomic search or a weighted linear regression. Once the solutions of these sub-problems are known, we can build the solution of the problem in linear time.

Equation (7) does not model the acceleration from cas_1 to cas_2. As a consequence, when solving the optimization problem defined by Equation (9b), we have only considered the points with an altitude superior to 13,000 ft, not 10,000 ft. This threshold choice is justified by Figure 6 where we can see the effect of the acceleration phase starting from 10,000 ft till 13,000 ft approximately.

On Figure 9, a climbing segment and the fitted speed profile are plotted. This climbing segment was selected among the climbing segments with at least 3 points above the crossover altitude and 3 points below $10{,}000$ ft. Among these segments, the climbing segment selected is the one with the median error. This error is computed with Equation (8). Thus, the speed profile accuracy in Figure 9 is quite representative of what can be obtained through this fitting process.

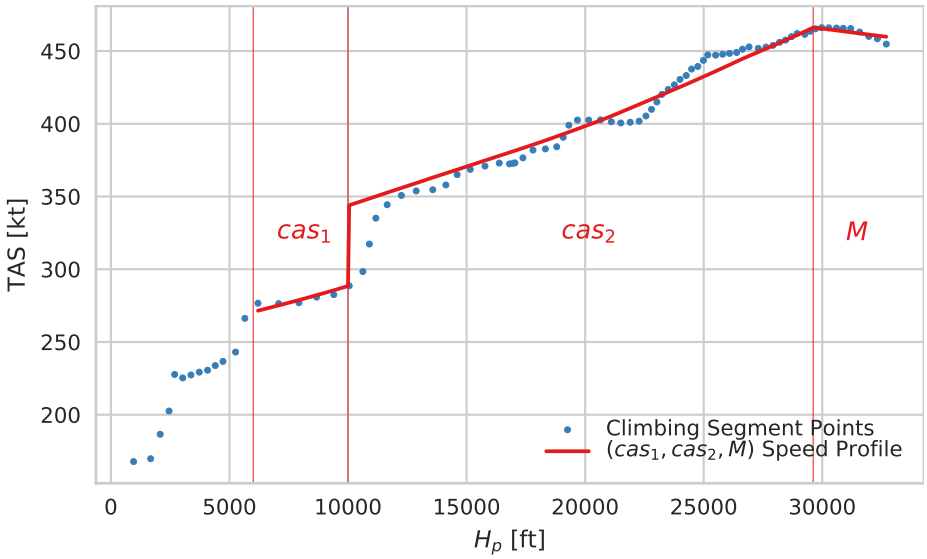


Figure 9: A climbing segment and the fitted (cas_1, cas_2, M) speed profile.

4.3. Explanatory Variables

The previous subsection describes how we obtained the target variables to be learned. In this subsection the explanatory variables used and the feature engineering done on these variables.

The explanatory variables contain information on the past positions and speeds. The explanatory variables include the altitude H_p, the vertical rate with respect to the ground $\frac{dh}{dt}$ and the True AirSpeed V_a of the current point and the 9 past points. Variables associated to the current point were added: the track angle, the ground speed, the latitude and the longitude.

They also include information on the weather: the temperature and wind at the current point, and the temperature every $1{,}000$ m starting from the current altitude H_p to the altitude $H_p + 11{,}000$ m. This is useful as the temperature does not follow a temperature profile corresponding to an ISA atmosphere even when this temperature profile is corrected with a temperature differential ΔT. The temperature at $H_p = 0$ was also added. The temperature can influence the engine performance. It also impacts the geopotential altitude H between two given geopotential pressure altitude Hp (see BADA user manual). It also impacts the speed profile. As a consequence, depending on the temperature gradient, different energy share factor will be used, hence different climbing rate.

For each example, we also have categorical variables like the airline operator, the aircraft type variant, the departure and arrival airports and the day of the week. This last variable was included to make use of a possible seasonality. The month can also provide some insight on the seasonality however our data only covers one year so the month was not included. All these categorical variables are handled by the gradient boosting tree library with a one-vs-other split algorithm.

When the departure and arrival airports were known we have computed the trip distance between these two airports. This trip distance will provide information on the fuel load and hence the mass of the aircraft which affects the climb. The departure airport is used because the constraints that apply to the climbing aircraft might depend on the airport considered.

We have also computed additional variables from the above variables. As these additional variables are computed from already available variables, these new variables do not add any information. However, they might be more relevant to the prediction problem we are dealing with. Thus, adding these new variables will help the learning algorithm to build a simple and accurate model. This kind of process is called *feature engineering*. As part of this *feature engineering*, we have added the mass estimated from the past points. The difference between the computed power using the mass and the observed energy variation was also added. This variable will give some hints on how reliable is the mass estimated on past points. We have also computed the energy variation on the past points using the formula: $g_0 \frac{T_i}{T_{ISA\,i}} \left(H_{p_{i+1}} - H_{p_i} \right) + \frac{1}{2} \left({V_{ai+1}}^2 - {V_{ai}}^2 \right)$. All these variables are related to the mass prediction.

In the end, we obtained 66 explanatory variables described in the table 4. All these variables are used as features by the $\text{BADA}_{\text{pred}}$ setup defined at the beginning of section 5. The 7 features available before take-off are used as features by the $\text{BADA}_{\text{pred-before-take-off}}$ setup defined at the beginning of section 5.

Table 4: A summary of the features used to predict the operational factors.

	feature description	number of features
before take-off	departure and arrival airports	2
	distance between airports	1
	aircraft type variant	1
	airline operator	1
	day of the week	1
	temperature at $H_p = 0$	1
with 9 past points	mass estimated on past points and error on past points	2
	track angle at the current point	1
	ground velocity at the current point	1
	north and east wind components	2
	longitude and latitude at the current point	2
	vertical speed at the current and past points	10
	Kalman filtered altitude H_p at the current and past points	10
	Kalman filtered airspeed V_a at the current and past points	10
	Kalman filtered energy variation between the current and past points	9
	temperature profile from current altitude to current altitude plus 11,000 m	12

4.4. Gradient Boosting Library

In this study we consider millions of climbing segments. We consider each aircraft type separately. Even with a separate model for each aircraft type, the A320 and the

B738 still contain over one million climbing segments, each. We have seen that one climbing segment generates several trajectory samples and hence several examples. For instance, the A320 training and validation sets contain almost 27 millions examples together. In a previous study (Alligier et al. (2015b)), the `R` package `gbm` was used. In this study, due to the large size of the training sets, its usage is impractical.

In our experiments we used the `LightGBM` package (Ke et al. (2017)), an open source Gradient Tree Boosting library developed by Microsoft. This implementation is highly efficient. Each continuous variable x_i is transformed into a discrete variable. To do so, the range of each variable x_i is split into bins. Each continuous value falls into a bin. The id of this bin will replace the continuous value. This allows the algorithm to reduce the memory usage. For instance, if each continuous range is split into 255 bins, then each `float64` value can be replaced by a `uint8` value. During the training process with our largest training set, `LightGBM` only used approximately 10 GB of RAM. The training process is also faster. During the tree build process, when the algorithm have to choose a splitting variable and a threshold, the algorithm only have 255 threshold values to consider for each variable.

As we consider a regression problem, we used the risk given by a quadratic loss $L(\hat{y}, y) = (\hat{y} - y)^2$. Let us note $\text{GBM}_{(m,J,\nu)}$ this algorithm, where m is the number of boosting iterations, J is the number of leaves of the tree and ν is the shrinkage parameter. The obtained model is a sum of regression trees. J allows us to control the interaction between variables, as we have $J-1$ variables at most in each regression tree. ν is the learning rate. The hyperparameters grid used for this algorithm is presented in Table 5. The number of boosting iterations m is determined using *early stopping*. With *early stopping*, the training process is stopped when the validation error does not improve for 20 iterations. If the training process has stopped at the round n, then the selected m is $n - 20$.

Table 5: Grid of hyperparameters used in our experiments.

method	hyperparameter grid
$\text{GBM}_{(m,J,\nu)}$	$m =$ determined by early stopping $J = \{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100,$ $110, 120, 130, 140, 150, 160, 170, 180, 190\}$ $\nu = \{0.02, 0.05\}$

4.5. Computing the Predicted Trajectory

Once we have predicted the operational factors like the mass and the (cas_1, cas_2, M) values, we have to compute the associated predicted trajectory. This is done by using the physical model BADA 3.14. This model specifies different mathematical functions used to compute the forces and the fuel consumption. Once these forces are computed, Newton's laws of motion can be used to compute the acceleration of the aircraft. The BADA model is a total energy model. With this model, the vertical motion is computed using the power of the forces and the energy share factor. This latter is used to specify how the aircraft is operated by the pilot. It is not rooted in the physical motion's laws.

The energy share factor e is defined by the equation (10).

$$e \overset{def.}{=} \frac{g_0 \frac{T}{T_{ISA}} \frac{dH_p}{dt}}{g_0 \frac{T}{T_{ISA}} \frac{dH_p}{dt} + V_a \frac{dV_a}{dt}} = \frac{1}{1 + \frac{V_a}{g_0} \frac{T_{ISA}}{T} \frac{dV_a}{dH_p}} \tag{10}$$

With the above equation, knowing the TAS V_a, the temperature T and the altitude H_p at the date t, we can compute the power $Power$ at t, the position and speed of the aircraft at $t + dt$ with the equation (11) below.

$$H_p(t + dt) = H_p + \frac{dH_p}{dt} dt = H_p + e \, \frac{T_{ISA}}{T} \frac{Power}{m g_0} dt \tag{11a}$$

$${V_a}^2(t + dt) = {V_a}^2 + \frac{1}{2}(1 - e) \frac{Power}{m} \, dt \tag{11b}$$

Starting with the predicted mass at the last observed altitude and speed, the predicted trajectory can be computed using the Euler method with $dt = 1 \; s$. The energy share factor e used is computed each dt considering the weather forecast and the (cas_1, cas_2, M) speed profile.

In (11), the energy share factor is required and not known. This energy share factor can be derived from (10) and the speed profile definition (9). If we assume an ISA atmosphere, one can derive an analytical formula (Mouillet (2017)) for the energy share factor. However, in this study, we take advantage of a temperature profile (one temperature measure every 1,000 m). Thus, the analytical formula cannot be used. This temperature profile does not follow an ISA atmosphere even if it is corrected by a ΔT temperature differential. The temperature profile is a function of the altitude H_p. Hence, if (cas_1, cas_2, M) is known, the speed profile is a function f of the altitude H_p. Using the equation (11a), the equation (11b) can be rewritten as below.

$$f^2 \left(H_p + e \, \frac{T_{ISA}}{T} \frac{Power}{m g_0} dt \right) = {V_a}^2 + \frac{1}{2}(1 - e) \frac{Power}{m} \, dt \tag{12}$$

In order to solve Equation 12, it is more convenient to consider a function g as defined by Equation (13) below.

$$g(e) = f^2 \left(H_p + e \, \frac{T_{ISA}}{T} \frac{Power}{m g_0} dt \right) - \left({V_a}^2 + \frac{1}{2}(1 - e) \frac{Power}{m} \, dt \right) \tag{13}$$

The equation $g(e) = 0$ is the one used to find the energy share factor e. We search for a value of e in the interval $[0.3, 1.7]$ as 0.3 and 1.7 are considered in BADA (Mouillet (2017)) as energy share factor values for acceleration and deceleration respectively.

Solving this equation is not straightforward as this function have discontinuities and might be not monotonic (see Figure 9). However, the discontinuities of g are the discontinuities of f. If we consider a continuous temperature profile, then only the discontinuity of the target speed remains. These discontinuity happens at known

altitudes. It is easy to find the discontinuities of g by solving $H_p + e\ \frac{T_{ISA}}{T}\frac{Power}{mg_0}dt = H_{p_{\text{discontinuity}}}$. Then if a discontinuity happens in $[0.3, 1.7]$, we can split the interval in order to have intervals where g is continuous.

Solving this equation on these continuous intervals is not easy as g might be not monotonic, so if the sign of g is the same at the boundaries of the interval that does not mean that the equation has no solution. To mitigate this issue, we split again each interval into intervals of size inferior to 0.1 and test if the sign of g at the boundaries of these smaller intervals are the same. By doing this, we increase the chance of detecting a sign change of g.

If all the evaluations of g are negative, we consider that there is no solution and we choose $e = 1.7$ as the aircraft needs to decelerate. Conversely, if all g are positive, we choose $e = 0.3$.

If two evaluations of g have a different sign then we apply a dichotomic search. This search stops when the search interval size is inferior to 0.001. This guarantees an error on the $\frac{dH_p}{dt}$ inferior to $0.1\ \%$ (compared with the value obtained with the exact e).

5. Results

All the results presented in this section have been computed on data not used in the model building process. These results have been computed from all the trajectories in the months November and December of the year 2017. The ten first months of 2017 were used to build the predictive models. The *training set* uses trajectories recorded from January to August, the *validation set* use trajectories recorded in September and October and finally the *test set* use trajectories recorded in November and December. It is a simple hold-out validation. The use of these three sets is described in subsection 2.2. One can refer to the Figure 3 of the subsection 2.2 for a schematic description of the training, validation and test sets.

The k-fold cross-validation usually provides a better assessment of the generalization error than a simple hold-out validation, nevertheless we chose the second approach here, for very specific reasons. The trajectories in the same area at the same time share the same local events: these trajectories are not independent. Using a cross-validation, these trajectories might fall in different folds and the performance evaluation might be optimistically biased. The distribution of the trajectories might change through time if for instance new procedures are applied at a specific airport. Using a cross-validation that randomly places the examples in the folds will produce folds with the same distribution. It will mask the non-stationarity of the problem we are studying and the performance evaluation obtained will be too optimistic. For these reasons, we chose a more practical approach, and decided that the model should be trained on a given period of time, and then tested on a later period of time, as would actually happen if the method was used in operations. If this performance evaluation is biased, it will be pessimistically biased.

We compute the trajectories using BADA 3.14. In this study, we compare four setups:

- $\text{BADA}_{\text{mean}}$ that uses BADA with the mean mass and mean values of cas_1, cas_2 and M;

- $\text{BADA}_{\text{mass}}$ that uses BADA with the mass estimated through the method presented in Alligier et al. (2014) and mean values of cas_1, cas_2 and M. It uses only the BADA physical model and the 9 past points of the considered aircraft;

- $\text{BADA}_{\text{pred-before-take-off}}$ that uses BADA with the mass, cas_1, cas_2 and M predicted by GBM. It uses only features that are available before the take-off of the considered aircraft;

- $\text{BADA}_{\text{pred}}$ that uses BADA with the mass, cas_1, cas_2 and M predicted by GBM. It uses all the available features available, including the 9 past points of the considered aircraft.

The mean parameters were computed using the ten first months. We do not use the BADA reference parameters as they were under-performing in several publications (Alligier et al. (2014, 2013)). Instead, we use the mean values. The mean values are the best we can do without any specific information. Machine Learning, if applied correctly, should perform better than the mean because the predictive models use specific information about the considered flight. We added the mass estimation method developed in Alligier et al. (2014) as it was demonstrated to perform well in several publications (Alligier et al. (2014, 2013); Uzun and Koyuncu (2017)). This estimation method uses the physical model and information about the motion of the considered aircraft. As opposed to the predictive models based on Machine Learning, it does not use past flights nor variables like the departure airport and the distance between the two airports. As a consequence the estimation method should perform better than the mean, but worse than the predictive models built through Machine Learning.

We study these setups in two different contexts:

- Past points are available: the aircraft is flying and we have at least 9 past points. In this context, the four setups can be used. Several situations are considered. The section 5.1 present the results obtained when predicting the operational factors. In the section 5.2, we consider the exact situation depicted by our training and validation sets. In section 5.3, we consider a 5 minutes prediction horizon. This situation differs from the one in our training set because the learned mass, cas_1, cas_2 and M were adjusted on a 10 minutes horizon, not a 5 minutes horizon. The section 5.4 studies also a 5 minutes horizon but on climbing segments smaller than the one used in our training set. These last three sections are here to be sure that our method can be successfully applied on any prediction horizon and any climbing trajectory.

- Past points are not available: the aircraft has not took-off yet, its past points are not available. Still, we want to predict the mass, cas_1, cas_2 and M. The results obtained in this context are discussed in 5.5. The setups that can be used in this context are compared: $\text{BADA}_{\text{mean}}$ and $\text{BADA}_{\text{pred-before-take-off}}$.

5.1. Prediction of the Operational Factors

The Table 6 presents statistics on the predicted mass for $\text{BADA}_{\text{mean}}$, $\text{BADA}_{\text{mass}}$ and $\text{BADA}_{\text{pred}}$. When compared with $\text{BADA}_{\text{mean}}$, $\text{BADA}_{\text{pred}}$ reduces the RMSE on the

mass by at least 42 % for all the aircraft types. The mean RMSE reduction among the aircraft type is 56 %. When compared with $\text{BADA}_{\text{mean}}$, the RMSE reduction provided by $\text{BADA}_{\text{mass}}$ is less obvious. Actually, with this method, the RMSE is higher than $\text{BADA}_{\text{mean}}$ at low altitude and lower at high altitude.

The estimated mass used in $\text{BADA}_{\text{mass}}$ is obtained from the method described in Alligier et al. (2014). It relies on the physical model and the aircraft motion on a few past points. These points are used to compute the energy variation. These estimated energy variation might be noisy. As a consequence the estimated mass is also noisy. For this reason, the mass actually used with $\text{BADA}_{\text{mass}}$ is $max(mass_{\text{min}}, min(mass_{\text{estimated}}, mass_{\text{max}}))$. In the Table 6, we computed the error made when we use the estimated mass without limiting the mass inside the $[mass_{\text{min}}; mass_{\text{max}}]$ interval. This setup is noted $\text{BADA}_{\text{mass-unlimited}}$ in this Table. Without limiting the mass, we have a higher RMSE for all the aircraft except the A332.

We have decided that the mass used with $\text{BADA}_{\text{pred}}$ is the predicted value, even if this value is outside $[mass_{\text{min}}; mass_{\text{max}}]$. The predicted mass is obtained from a predictive model. This model is built using a set of examples. This set of examples is built by targeting the mass estimated on the future points. This estimation is less noisy than the estimated mass because it uses more points of the sets of examples. Furthermore, this mass estimated on future points is directly related to the mean energy variation observed in the future points and is consequently suitable for predicting the future trajectory. These masses are used to train predictive models using Machine Learning. The GBM Machine Learning method computes tree regression models that basically just computes mean values of the learned mass. Thus, if a predicted mass is outside $[mass_{\text{min}}; mass_{\text{max}}]$ that means that several examples with a similar x were associated with a mass outside $[mass_{\text{min}}; mass_{\text{max}}]$.

Table 6: These statistics, in kg, are computed on the difference between the predicted mass and the mass extracted from the future trajectory. It is computed on the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$		$\text{BADA}_{\text{mass-unlimited}}$	
statistic	mean	rmse	mean	rmse	mean	rmse	mean	rmse
B738	132	5,212	3,464	5,774	−70	2,508	4,145	7,555
A320	92	5,065	1,360	3,504	−91	1,929	1,698	4,732
A319	110	4,810	2,360	5,048	−131	2,362	2,901	7,108
A321	207	6,699	1,236	4,222	−105	2,212	3,105	5,181
E195	187	4,393	1,274	3,343	−235	2,126	2,433	4,383
E190	−569	4,418	1,798	3,620	−702	2,539	2,727	5,147
DH8D	425	1,811	−937	2,172	−48	738	−881	2,281
B737	96	4,842	4,126	6,148	−75	2,511	5,304	8,386
CRJ9	−192	2,818	60	1,943	−110	1,294	345	2,239
A332	920	26,910	−6,506	16,339	−627	8,014	−783	12,764
B77W	−917	30,105	15,937	28,456	−1,913	10,742	17,951	32,341

The Table 7 presents statistics on the predicted cas_1 for $\text{BADA}_{\text{mean}}$ and $\text{BADA}_{\text{pred}}$. The RMSE reduction for these parameters is smaller than the one observed for the

mass. The mean RMSE reduction is 49 %. We have also computed the statistics on the predicted cas_2 on the Table 8. On this operational factor, the mean RMSE reduction is 39 %.

Table 7: These statistics, in kt, are computed on the difference between the predicted cas_1 and the value extracted from the future trajectory. It is computed on the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse
B738	−1.13	18.34	0.08	8.44
A320	−1.62	18.97	0.07	10.10
A319	0.05	16.49	0.26	8.31
A321	−0.31	14.56	0.07	8.42
E195	0.85	11.87	0.19	7.59
E190	0.44	14.15	0.56	8.22
DH8D	−1.49	19.08	0.13	6.95
B737	0.13	15.77	0.04	6.47
CRJ9	0.73	16.19	−0.20	9.10
A332	−0.12	15.60	0.25	9.66
B77W	0.94	19.54	−0.31	7.11

Table 8: These statistics, in kt, are computed on the difference between the predicted cas_2 and the value extracted from the future trajectory. It is computed on the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse
B738	−0.21	13.54	−0.31	7.84
A320	0.04	14.89	−0.24	8.85
A319	−0.08	15.89	−0.37	9.50
A321	−0.51	15.83	−0.57	9.48
E195	−0.71	11.00	−0.68	7.45
E190	0.65	11.72	0.02	7.09
DH8D	−5.00	18.40	−0.19	11.53
B737	0.37	13.85	−0.20	8.20
CRJ9	−2.04	14.59	−0.25	7.44
A332	0.73	13.10	−0.20	7.35
B77W	−0.16	7.07	−0.09	5.47

The Table 9 presents statistics on the predicted Mach number M for $\text{BADA}_{\text{mean}}$ and $\text{BADA}_{\text{pred}}$. The RMSE reduction on this parameter is very small, the mean RMSE reduction is only 15 %. For the DH8D, $\text{BADA}_{\text{mean}}$ has a lower RMSE than $\text{BADA}_{\text{pred}}$. According to the BADA model, the Mach number parameter impacts the trajectory only at high altitude. As a consequence, the past points of the trajectory (with an al-

titude typically inferior to $30{,}000$ ft as can be seen on Figures 10 and 11) might not contain useful information to predict the Mach number. As opposed to the Mach number, in the BADA model, the mass parameter has an impact on the whole trajectory not just a part of it. This might explain the performance gap between the model predicting the mass and the one predicting the Mach number.

Table 9: These statistics are computed on the difference between the predicted Mach number M and the value extracted from the future trajectory. It is computed on the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse
B738	1.584e-05	0.01761	-0.0004014	0.01476
A320	0.0004234	0.02375	-0.0001622	0.0197
A319	0.0002773	0.02058	-0.0001642	0.0174
A321	0.000483	0.02059	-0.0005212	0.01804
E195	-0.001265	0.0301	-0.001155	0.02495
E190	0.0007452	0.0228	0.0003135	0.01993
DH8D	-0.01177	0.0349	-0.002219	0.02357
B737	0.0005285	0.01472	-0.0002208	0.01305
CRJ9	-0.0003177	0.02254	7.63e-05	0.0202
A332	0.001277	0.02594	-0.0004583	0.0229
B77W	-0.001055	0.01692	-0.0005447	0.01532

5.2. Trajectory Prediction at t_0+10 minutes

We compare three different setups for the operational factors as explained at the beginning of the section 5. The predicted trajectories are computed using the predicted operational factors and the the method described in section 4.5.

5.2.1. Altitude

Table 10 presents statistics on the error made on the altitude at a 10 minutes horizon. We can see that $\text{BADA}_{\text{mean}}$ does not provide a prediction with a mean error close to zero, especially for the DH8D. This can be explained by the fact that we do not have $E[f(X)] = f(E[X])$ for every function f. If f is the BADA process computing the altitude at $t_0 + 10$ min and X is the input parameter, it is easy to understand that even with the mean mass and the mean (cas_1, cas_2, M) we do not obtain a zero mean error on the altitude. When compared with the baseline $\text{BADA}_{\text{mean}}$, the use of the predicted mass and speed profile reduce the Root Mean Squared error (RMSE) on the altitude by at least 38 % for all the considered aircraft types. The mean error reduction among the aircraft type is 48 %. The largest reduction is obtained with the A332 for which we have a 64 % reduction on the RMSE.

The relative reduction obtained in this study is similar to the one obtained in Alligier et al. (2015b). However, the RMSE values obtained in this study with $\text{BADA}_{\text{pred}}$ are larger than the one obtained in Alligier et al. (2015b). This can be explained by

Table 10: These statistics, in feet, are computed on the differences between the predicted altitude and the observed altitude $\left({H_p}^{(pred)} - {H_p}^{(obs)}\right)$ at the time $t = 600$ s for the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse
B738	−227	1,758	−1,114	1,929	−115	930
A320	−138	1,779	−409	1,355	−59	886
A319	−160	1,719	−577	1,616	−43	975
A321	−169	1,936	−339	1,359	−27	846
E190	−75	1,854	−940	1,789	42	1,149
E195	−260	1,742	−603	1,485	−109	1,008
DH8D	503	1,756	1,691	2,883	334	1,044
B737	−328	1,786	−1,716	2,505	−246	1,078
CRJ9	13	1,731	−43	1,329	−30	911
A332	−196	2,440	559	1,499	62	871
B77W	−237	2,541	−1,328	2,402	−21	948

the fact that this study considers every trajectory samples, not just the one departing from Paris Charles de Gaulle (LFPG) with $H_p(t = 0) \geqslant 18{,}000$ ft like in Alligier et al. (2015b). The altitude at $t = 0$ has a great impact on the uncertainty of the altitude at $t = 600$ s. The higher the starting altitude is, the lower is the uncertainty on the final altitude. It might also be explained by the variety of the airports and airspace considered. Actually, if we consider only aircraft departing from LFPG with $H_p(t = 0) \geqslant 18{,}000$ ft then we obtain results comparable with the one obtained in Alligier et al. (2015b).

Figures 10 and 11 plot the error on the altitude as a function of the starting altitude for different aircraft types. On Figure 10, we have some aircraft types for which $\text{BADA}_{\text{mass}}$ provides poor results. This is especially true for the DH8D and B737. No explanation was found for this behavior, yet. Alligier et al. (2015a) have also demonstrated that the mass estimation method used in $\text{BADA}_{\text{mass}}$ performed poorly on the B737. The other aircraft types were plotted in Figure 11. For the B738 and the B77W, the mass estimation performs way better at high altitude than at low altitude.

We can see that $\text{BADA}_{\text{pred}}$ performs consistently better than all the other methods, for all the altitude and aircraft types considered.

5.2.2. Results for All the Airports Considered

Figure 12 presents the RMSE percentage reduction for all the airports. In this figure, we have one boxplot for each aircraft type. Considering a given aircraft type, for each airport with more than 100 climbing segments we have computed the RMSE of $\left({H_p}^{(pred)} - {H_p}^{(obs)}\right)$ $(t = 600\ s)$ for the baseline method $\text{BADA}_{\text{mean}}$ and our method $\text{BADA}_{\text{pred}}$. The percentage reduction computed for the considered airport gives one point on the figure. A boxplot is associated to the points of each considered aircraft type. To summarize, for each aircraft type, one airport correspond to one point, and the position of this point is the RMSE reduction on the altitude on a 10 minutes prediction

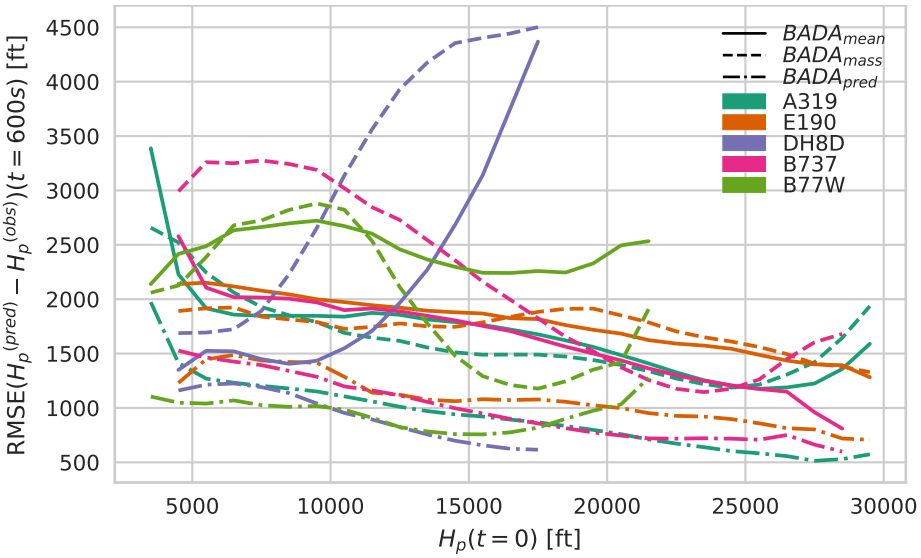


Figure 10: RMSE of $\left(H_p{}^{(pred)} - H_p{}^{(obs)}\right)$ $(t = 600\ s)$ as a function of the starting altitude. The aircraft types plotted are the one for which $\text{BADA}_{\text{mass}}$ provides poor results.

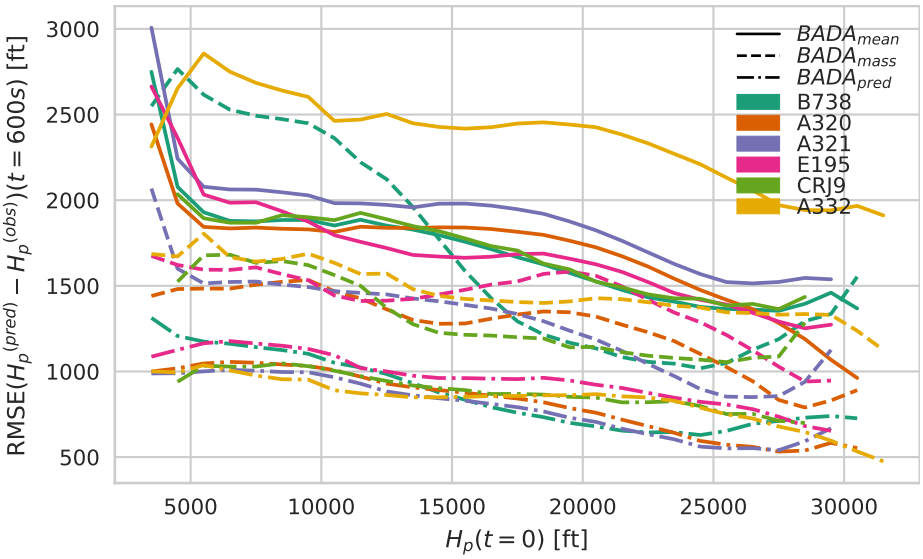


Figure 11: RMSE of $\left(H_p{}^{(pred)} - H_p{}^{(obs)}\right)$ $(t = 600\ s)$ as a function of the starting altitude. The aircraft types plotted are the one for which $\text{BADA}_{\text{mass}}$ provides good results.

horizon.

With this figure, we can see that we have at least a 15 % RMSE reduction for all the airports and aircraft types considered. For the aircraft types with a lot of airports, using the boxplots, we can see that half of the airports have a RMSE reduction within $\pm$ 7 % around the median.

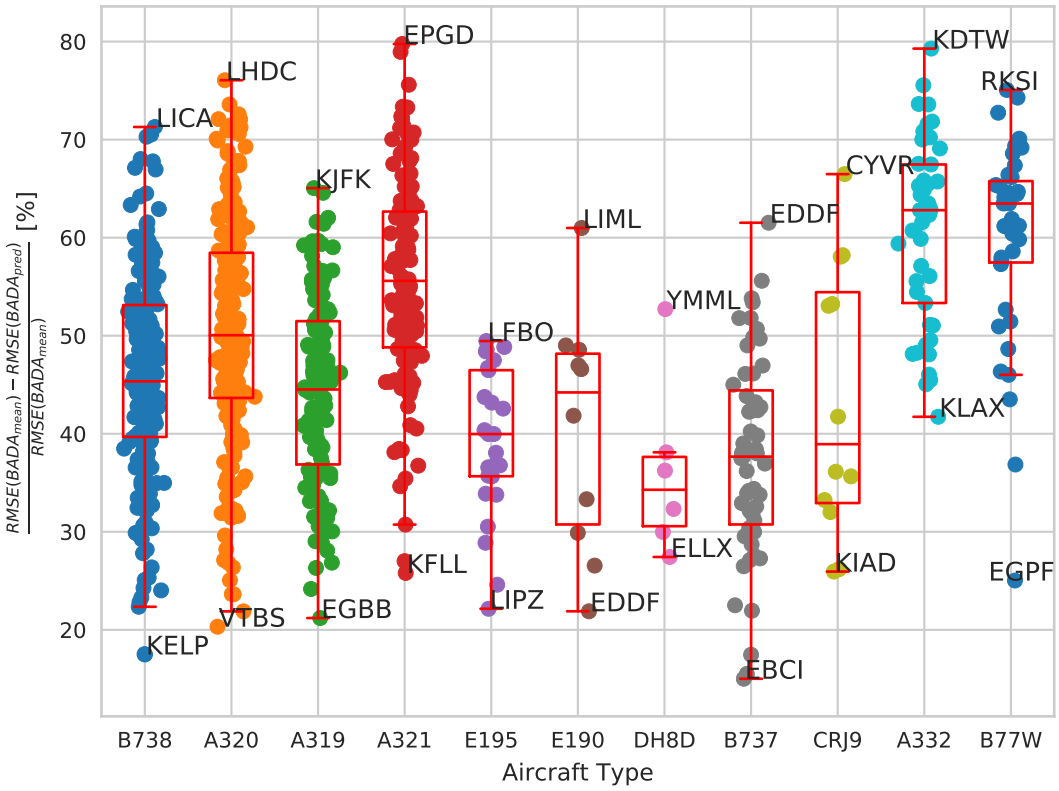


Figure 12: For each aircraft type, one airport correspond to one point, and the position of this point is the RMSE reduction on the altitude on a 10 minutes prediction horizon. Only the airports with more than 100 climbing segments in the *test set* has been plotted.

5.2.3. Speed

Table 11 presents statistics on the error made on the speed within the ten minutes time window. This error is computed at corresponding time.

When compared with the baseline $\text{BADA}_{\text{mean}}$, the use of the predicted mass and speed profile in $\text{BADA}_{\text{pred}}$ reduces the RMSE on the speed by at least 15 % for all the considered aircraft types. The largest reduction is obtained with the DH8D for which we have a 35 % reduction on the RMSE. The relative reduction obtained in this study is worse than the one obtained in Alligier et al. (2015b). However, the RMSE values

obtained in this study with $\text{BADA}_{\text{pred}}$ are similar to the one obtained in Alligier et al. (2015b).

Despite having the same (cas_1, cas_2, M) values, the RMSE computed using $\text{BADA}_{\text{mean}}$ and $\text{BADA}_{\text{mass}}$ are not the same. This comes from two facts: the (cas_1, cas_2, M) describes the speed profile as a function of the altitude H_p; the RMSE is computed between speed at corresponding time not corresponding altitude. Hence, if the mass used is different, the rate of climb will be different and the altitude at a given time will be also different. However, as can be seen in Table 11, $\text{BADA}_{\text{mass}}$ does not provide any improvement on the speed prediction.

Table 11: These statistics, in kt, are computed on the differences between the predicted TAS and the observed TAS on $\left(V_a{}^{(pred)} - V_a{}^{(obs)}\right)$ at the time $0 \leqslant t \leqslant 600$ s for the *test set* consisting in trajectory samples from months November and December.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse
B738	1.28	16.12	−2.37	16.26	0.56	12.16
A320	1.16	17.00	0.07	16.92	0.69	13.15
A319	0.93	17.58	−0.75	17.36	0.53	13.55
A321	1.26	17.86	0.17	17.69	0.75	13.95
E190	2.93	15.98	−0.20	14.85	2.01	11.72
E195	0.39	15.17	−1.03	14.56	0.78	11.22
DH8D	−5.77	21.07	−5.54	21.29	0.04	13.81
B737	2.39	15.96	−1.68	15.94	1.07	12.10
CRJ9	−0.26	15.73	−0.31	16.11	0.74	11.62
A332	0.63	16.17	2.41	16.62	0.35	12.49
B77W	1.20	15.63	−3.70	15.60	1.72	13.03

5.3. Trajectory Prediction at t_0+5 minutes

The masses in our set of examples were adjusted on a 10 minutes climb portion. The adjusted mass basically fits the mean energy variation for the 10 minutes climb portion. If the physical model is not well designed, the model used with the adjusted mass might overestimate the energy variation for the first 5 minutes and understimate it for the last 5 minutes. This subsection investigates how our models perform on a 5 minutes prediction horizon.

With Table 12, we can see that, when compared with $\text{BADA}_{\text{mean}}$, $\text{BADA}_{\text{pred}}$ reduces the RMSE on the predicted altitude at a 5 minutes horizon by at least 30 %, except for the DH8D. For the DH8D, almost no reduction was observed. For the A321, it is even reduced by 50 %.

5.4. Trajectory Prediction on Smaller Climbing Segments

In this subsection we test if the Machine Learning predictive models are efficient when we consider small climb segments. This is not trivial as the Machine Learning predictive models were trained on examples built from climbing segments longer than

Table 12: These statistics, in feet, are computed on the differences between the predicted altitude and the observed altitude $\left({H_p}^{(pred)} - {H_p}^{(obs)}\right)$ at the time $t = 300$ s for the *test set* consisting in trajectory samples from months November and December. Only climbing segments with a duration superior to 750 s were considered.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse
B738	−227	1,276	−384	1,065	−92	718
A320	−178	1,279	−159	957	−4	681
A319	−208	1,280	−190	1,114	−214	883
A321	−177	1,381	−73	935	35	660
E190	−124	1,376	−310	1,067	137	848
E195	−255	1,321	−175	978	114	720
DH8D	−26	1,200	753	1,571	−889	1,177
B737	−351	1,378	−587	1,246	−162	871
CRJ9	−127	1,223	−11	906	−50	667
A332	−293	1,657	213	968	−5	622
B77W	−338	1,707	−505	1,132	−210	1,008

750 s. Hence, if we use it on smaller climbing segments, we have no guarantee that it will perform well. Intuitively, the pilot might operate his aircraft in a completely different way depending on the planned climb duration. With our models, we have learned how the pilot operates when the duration of the climb is superior to 750 s. Strictly speaking, it does not tell us how he or she operates the climb for smaller segment. Actually, these smaller segments were not part of our training set.

In a more formal way, we could have two distributions $p_{\text{small}}(x, y)$ and $p_{\text{long}}(x, y)$ that might be very different where x are the explanatory variables (*i.e.* the past points) and y the future trajectory. $p_{\text{small}}(x, y)$ models the distribution for small climbing segments and $p_{\text{long}}(x, y)$ the distribution for long climbing segments. Given similar past points x, the pilot could operates differently depending on the planed duration, that would lead us to have $p_{\text{small}}(y|x) \neq p_{\text{long}}(y|x)$. Even if the conditional distribution were the same, we could have $p_{\text{small}}(x) \neq p_{\text{long}}(x)$. In this case, we might have a *covariate shift* (Shimodaira (2000)) between a *training set* containing long climbing segments and a *test set* containing short climbing segments. This will lead the model to perform differently when used on smaller climbing segments. As we can see, testing our models on smaller climbing segments will give information that are not available in the previous subsection.

Table 13 presents the results obtained when we consider climbing segments with a duration strictly inferior to 750 s. On these segments, we have tried to predict the altitude at $t_0 + 5$ min. On these short climbing segments, we can see that $\text{BADA}_{\text{pred}}$ performs better than the other methods except for the DH8D. However, the values of RMSE obtained are larger than the one obtained on longer segments as presented in the Table 12.

Table 14 presents the results obtained when we consider climbing segments with a duration strictly inferior to 450 s. On these segments, we have tried to predict the

altitude at $t_0 + 2.5$ min. We can see that $\text{BADA}_{\text{pred}}$ performs better than the other methods except for the DH8D and the B77W for which $\text{BADA}_{\text{mass}}$ performs better. However, the gap between $\text{BADA}_{\text{pred}}$ and $\text{BADA}_{\text{mass}}$ is small. For the DH8D, the RMSE is even reduced by 25 % when $\text{BADA}_{\text{mass}}$ is used over $\text{BADA}_{\text{pred}}$.

Table 13: These statistics, in feet, are computed on the differences between the predicted altitude and the observed altitude $\left({H_p}^{(pred)} - {H_p}^{(obs)}\right)$ at the time $t = 300$ s for the *test set* consisting in trajectory samples from months November and December. Only climbing segments with a duration strictly inferior to 750 s were considered.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse
B738	−646	1,617	−683	1,378	−209	927
A320	−407	1,477	−130	1,098	−78	848
A319	−600	1,580	−318	1,306	−398	1,124
A321	−450	1,525	−151	998	−14	789
E190	−385	1,591	−140	1,255	91	1,016
E195	−866	1,686	−266	1,141	−45	890
DH8D	−1,136	1,708	256	1,447	−1,404	1,736
B737	−860	1,722	−921	1,620	−296	1,023
CRJ9	−439	1,279	−69	995	−160	800
A332	−887	1,954	61	985	−73	753
B77W	−1,172	2,361	−801	1,494	−255	1,295

Table 14: These statistics, in feet, are computed on the differences between the predicted altitude and the observed altitude $\left({H_p}^{(pred)} - {H_p}^{(obs)}\right)$ at the time $t = 150$ s for the *test set* consisting in trajectory samples from months November and December. Only climbing segments with a duration strictly inferior to 450 s were considered.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse
B738	−268	1,089	−275	898	−116	760
A320	−223	1,008	−49	836	−108	705
A319	−228	1,155	−73	1,034	−324	998
A321	−312	1,021	−115	780	−85	640
E190	−183	1,008	−54	885	80	764
E195	−431	1,048	−38	849	−20	694
DH8D	−711	1,222	75	913	−1,071	1,302
B737	−443	1,171	−374	1,015	−197	862
CRJ9	−323	881	−82	794	−207	662
A332	−436	1,173	1	730	−96	620
B77W	−492	1,384	−290	939	−67	991

5.5. Trajectory Prediction Before Take-Off

In all the above subsection, we used past points of the trajectory to predict the operational factors like the mass and the (cas_1, cas_2, M) values. The methods $\text{BADA}_{\text{mass}}$ and $\text{BADA}_{\text{pred}}$ both required to observe some trajectory points. As a consequence the considered aircraft has already took-off.

In this subsection, we test predictive models that predict the operational factors, like the mass and the (cas_1, cas_2, M) values, using only information available long before the aircraft ever took-off. To do this, instead of using the 66 explanatory variables used in the previous sections, we will use only 7 variables containing information available before the aircraft take-off: the departure and arrival airports, the distance between these airports, the type variant, the airline operator, the day of week and the temperature at $H_p = 0$. Using these 7 explanatory variables also described in Table 4 and gradient tree boosting, we learn models to predict the operational factors.

In order to test the validity of the predicted operational factors, they will be tested on the trajectory samples used in 5.2. Of course, the temperature profile, the starting altitude and speed were used to compute the predicted trajectory. But none of these information were used to predict the operational factors. In the following, the method is noted $\text{BADA}_{\text{pred-before-take-off}}$. With Table 15, we can see that $\text{BADA}_{\text{pred-before-take-off}}$ performs better than $\text{BADA}_{\text{mean}}$ except for the DH8D. It also performs better than $\text{BADA}_{\text{mass}}$ except for the A321 and CRJ9. This is a good result as these methods do not require the same amount of information. $\text{BADA}_{\text{mass}}$ requires to have information on the aircraft past points whereas $\text{BADA}_{\text{pred-before-take-off}}$ requires only information available before the aircraft takes off.

Table 15: These statistics, in feet, are computed on the differences between the predicted altitude and the observed altitude $\left(H_p{}^{(pred)} - H_p{}^{(obs)}\right)$ at the time $t = 600$ s for the *test set* consisting in trajectory samples from months November and December. The used mass and (cas_1, cas_2, M) values were derived from information available before take-off.

method	$\text{BADA}_{\text{mean}}$		$\text{BADA}_{\text{pred-take-off}}$		$\text{BADA}_{\text{mass}}$		$\text{BADA}_{\text{pred}}$	
statistic	mean	rmse	mean	rmse	mean	rmse	mean	rmse
B738	−227	1,758	−70	1,278	−1,114	1,929	−115	930
A320	−138	1,779	26	1,352	−409	1,355	−59	886
A319	−160	1,719	27	1,362	−577	1,616	−43	975
A321	−169	1,936	59	1,393	−339	1,359	−27	846
E190	−75	1,854	272	1,566	−940	1,789	42	1,149
E195	−260	1,742	133	1,468	−603	1,485	−109	1,008
DH8D	503	1,756	564	1,762	1,691	2,883	334	1,044
B737	−328	1,786	−143	1,427	−1,716	2,505	−246	1,078
CRJ9	13	1,731	69	1,364	−43	1,329	−30	911
A332	−196	2,440	178	1,379	559	1,499	62	871
B77W	−237	2,541	41	1,261	−1,328	2,402	−21	948

Conclusion

To conclude, let us summarize our approach and findings, before giving a few perspectives on future works.

In this study we have tested Machine Learning methods using millions of climbing segments coming from The OpenSky Network. These climbing segments were completed with weather forecasts, aircraft types and variants, departure and arrival airports, estimated masses and speed profiles. The filtered and augmented data set is available at https://opensky-network.org/datasets/publication-data. The Machine Learning code is also available at the author's GitHub page https://github.com/richardalligier/trc2018. Inside the ATM trajectory prediction community, we hope that sharing the data set and the Machine Learning code will enable scientifically sound comparisons based on the exact same data set.

Using this data set, we applied a Machine Learning technique namely Gradient Boosting Machines, to learn predictive models. These models predict the mass and the (cas_1, cas_2, M) speed profiles values. The techniques used are similar to the one used in previous publications (Alligier et al. (2015b,a)). However, this study extends the latter in many ways. Several possible biases were removed. The climbing segments were not filtered according to their altitude. This resulted in the addition of the cas_1 parameter. The previous studies considered a limited number of climbing segments with one to two departure airports, a limited time period and altitude filtered segments. In this study, we consider trajectories from the whole year 2017 and from all over the world. This study worked with the ten most frequent aircraft types according to Centre (2017). These aircraft types cover 63 % of the European air traffic in 2016. The algorithms were applied to millions of climbing segments demonstrating that the Machine Learning techniques that we used can scale to this amount of data. In addition, this study demonstrated that the obtained predictive models can be used on small climbing segments and shorter prediction time horizons. It has also demonstrated that we can improve trajectory prediction using only information available before the aircraft even took-off.

Considering a 10 minutes prediction horizon, compared with mean values, the Machine Learning methods reduced the RMSE on the altitude by at least 35 % for all the considered aircraft. The largest reduction is obtained with the A332 for which we have a 64 % reduction on the RMSE. We have demonstrated that, unlike mass estimation, our predictive models provide a consistent reduction of the RMSE for all the starting altitudes and aircraft types considered. Our predictive models also improved the speed prediction, with an RMSE reduced by at least 15 % for all the aircraft considered. On a 5 minutes prediction horizon, for the altitude prediction, RMSE is reduced by at least 30 % for all aircraft types except DH8D. It also provides an improvement, albeit smaller, on even shorter climbing segments with a 2.5 minutes prediction horizon. This is not a trivial result as no short climbing segments were in the set of examples used to train the models. We have also demonstrated that our predictive models can predict accurate operational factors using only information available before the aircraft even took-off. Using only this information, the predicted parameters reduced the RMSE on the altitude by at least 15 % except for the DH8D. The largest reduction is obtained with the B77W for which we have a 50 % reduction of the RMSE.

From an operational point of view, the resulting improvement in the climb prediction accuracy would certainly benefit air traffic controllers, especially in the vertical separation task as shown in Thipphavong et al. (2012). Furthermore, being able to have accurate operational factors before the aircraft even took-off might help to reduce the along track error and the Top Of Climb prediction error.

For future work, it could be interesting to have a deeper look at the DH8D or at the airports for which the error is higher than the average. It could help us to identify the current model limitations and guide us towards helpful model modifications.

Acknowledgment

We thank The OpenSky Network team who granted us access to their wonderful database. We are also grateful that they host the data set that has been filtered and augmented with our algorithms so others can work easily on the same data set.

ADAPT2. aircraft data aiming at predicting the trajectory. data analysis report. Technical report, EUROCONTROL Experimental Center, 2009.

Richard Alligier, David Gianazza, and Nicolas Durand. Learning the aircraft mass and thrust to improve the ground-based trajectory prediction of climbing flights. *Transportation Research Part C: Emerging Technologies*, 36:45 – 60, 2013. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2013.08.006. URL `http://www.sciencedirect.com/science/article/pii/S0968090X13001708`.

Richard Alligier, David Gianazza, Mohammad Ghasemi Hamed, and Nicolas Durand. Comparison of Two Ground-based Mass Estimation Methods on Real Data (regular paper). In *International Conference on Research in Air Transportation (ICRAT), Istambul, 26/05/2014-30/05/2014*, page (on line), http://www.icrat.org, mai 2014. ICRAT. URL `www.icrat.org`.

Richard Alligier, D. Gianazza, and N. Durand. Machine learning and mass estimation methods for ground-based aircraft climb prediction. *Intelligent Transportation Systems, IEEE Transactions on*, 16(6):3138–3149, Dec 2015a. ISSN 1524-9050. doi: 10.1109/TITS.2015.2437452.

Richard Alligier, David Gianazza, and Nicolas Durand. Machine learning applied to airspeed prediction during climb. In *Proceedings of the 11th USA/Europe Air Traffic Management R & D Seminar*, Lisbon, Portugal, 2015b.

Christopher M Bishop. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

Marc Bourgois and Michael Sfyroeras. Open data for air transport research: Dream or reality? In *Proceedings of The International Symposium on Open Collaboration*, page 17. ACM, 2014.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984.

Jesper Bronsvoort, Greg McDonald, Mike Paglione, Christina M Young, Jean Boucquey, Joachim K Hochwarth, and Eduardo Gallo. Real-time trajectory predictor calibration through extended projected profile down-link. In *Eleventh USA/Europe Air Traffic Management Research and Development Seminar*, 2015.

Robert Grover Brown, Patrick YC Hwang, et al. *Introduction to random signals and applied Kalman filtering*, volume 3. Wiley New York, 1992.

EUROCONTROL Experimental Centre. Coverage of 2016 european air traffic for the base of aircraft data (bada) versions 3.14 & 4.2. Technical report, EUROCONTROL, June 2017.

Georgios Chaloulos, Eva Crück, and John Lygeros. A simulation based study of subliminal control for air traffic management. *Transportation Research Part C: Emerging Technologies*, 18(6):963 – 974, 2010. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2010.03.002. URL http://www.sciencedirect.com/science/article/pii/S0968090X10000288. Special issue on Transportation Simulation Advances in Air Transportation Research.

Yashovardhan S Chati and Hamsa Balakrishnan. Statistical modeling of aircraft takeoff weight. 2017.

SESAR Consortium. Milestone Deliverable D3: The ATM Target Concept. Technical report, 2007.

Richard A. Coppenbarger. Climb trajectory prediction enhancement using airline flight-planning information. In *AIAA Guidance, Navigation, and Control Conference*, 1999.

Fabrice Drogoul, Philippe Averty, and Rosa Weber. Erasmus strategic deconfliction to benefit sesar. In *Proceedings of the 8th USA/Europe Air Traffic Management R&D Seminar*, June-July 2009. URL http://atm2003.eurocontrol.fr/8th-seminar-united-states-june-2009.

Nicolas Durand, Jean-Marc Alliot, and Joseph Noailles. Automatic aircraft conflict resolution using genetic algorithms. In *Proceedings of the Symposium on Applied Computing, Philadelphia*. ACM, 1996.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics Data Analysis*, 38(4):367 – 378, 2002. ISSN 0167-9473. doi: http://dx.doi.org/10.1016/S0167-9473(01)00065-2. URL http://www.sciencedirect.com/science/article/pii/S0167947301000652.

Mohammad Ghasemi Hamed. *Méthodes non-paramétriques pour la prévision d'intervalles avec haut niveau de confiance: application à la prévision de trajectoires d'avions*. PhD thesis, Thèse doctorat informatique de l'INPT, 2014.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

Marko Hrastovec and Franc Solina. Machine learning model for aircraft performances. In *Digital Avionics Systems Conference (DASC), 2014 IEEE/AIAA 33rd*, pages 8C4–1. IEEE, 2014.

Marko Hrastovec and Franc Solina. Prediction of aircraft performances based on data collected by air traffic control centers. *Transportation Research Part C: Emerging Technologies*, 73:167 – 182, 2016. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2016.10.018. URL `http://www.sciencedirect.com/science/article/pii/S0968090X1630211X`.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3149–3157. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf`.

Rainer Koelle. Open source software and crowd sourced data for operational performance analysis. In *Proceedings of the 12th USA/Europe Air Traffic Management R & D Seminar*, Seattle, Washington, 2017.

Yann Le Fablec. *Prévision de trajectoires d'avions par réseaux de neurones*. PhD thesis, Thèse doctorat informatique de l'INPT, 1999.

Javier Lopes-Leonés. *The Aircraft Intent Description Language*. PhD thesis, University of Glasgow, 2007.

Javier López-Leonés, Miguel A. Vilaplana, Edouardo Gallo, Francisco A. Navarro, and Carlos Querejeta. The aircraft intent description language: A key enabler for air-ground synchronization in trajectory-based operations. In *Proceedings of the 26th IEEE/AIAA Digital Avionics Systems Conference*. DASC, 2007.

Peter Martin and Georges Mykoniatis. Study of the acquisition of data from aircraft operators to aid trajectory prediction calculation. Technical report, EUROCONTROL Experimental Center, 1998.

Vincent Mouillet. User manual for base of aircraft data (bada) rev.3.14. Technical report, EUROCONTROL, 2017.

Young S. Park and David P. Thipphavong. Performance of an Adaptive Trajectory Prediction Algorithm for Climbing Aircraft. In *2013 Aviation Technology, Integration, and Operations Conference*, page (on line). 08, Aug 2013. URL `http://dx.doi.org/10.2514/6.2013-4263`.

Xavier Prats, Vincenç Puig, Joseba Quevedo, and Fatiha Nejjari. Multi-objective optimisation for aircraft departure trajectories minimising noise annoyance. *Transportation Research Part C*, 18(6):975–989, 2010. URL `www.elsevier.com/locate/trc`.

Charles A Schultz, David Thipphavong, and Heinz Erzberger. Adaptive trajectory prediction algorithm for climbing flights. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, August 2012. ISBN 978-1-60086-938-9.

Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. Bringing Up OpenSky: A Large-scale ADS-B Sensor Network for Research. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, IPSN '14, pages 83–94, Berlin, Germany, April 2014.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2): 227–244, 2000.

Junzi Sun. World aircraft database [data file], 2017. Retrieved from http://junzis.com/adb/data.

Junzi Sun, Joost Ellerbroek, and Jacco Hoekstra. Modeling and inferring aircraft takeoff mass from runway ads-b data. In *7th International Conference on Research in Air Transportation*, 2016.

Junzi Sun, Joost Ellerbroek, and Jacco Hoekstra. Bayesian inference of aircraft initial mass. In *Proceedings of the 12th USA/Europe Air Traffic Management Research and Development Seminar*. FAA/EUROCONTROL, 2017a.

Junzi Sun, Joost Ellerbroek, and Jacco Hoekstra. Modeling aircraft performance parameters with open ads-b data. In *Proceedings of the 12th USA/Europe Air Traffic Management Research and Development Seminar*. FAA/EUROCONTROL, 2017b.

Harry Swenson, Richard Barhydt, and Michael Landis. Next Generation Air Transportation System (NGATS) Air Traffic Management (ATM)-Airspace Project. Technical report, National Aeronautics and Space Administration, 2006.

Kairat Tastambekov, Stéphane Puechmorel, Daniel Delahaye, and Christophe Rabut. Aircraft trajectory forecasting using local functional regression in sobolev space. *Transportation Research Part C: Emerging Technologies*, 39(0):1 – 22, 2014. ISSN 0968-090X. doi: http://dx.doi.org/10.1016/j.trc.2013.11.013. URL `http://www.sciencedirect.com/science/article/pii/S0968090X13002416`.

David P Thipphavong, Charles A Schultz, Alan G Lee, and Steven H Chan. Adaptive algorithm to improve trajectory prediction accuracy of climbing aircraft. *Journal of Guidance, Control, and Dynamics*, 36(1):15–24, 2012.

Mevlüt Uzun and Emre Koyuncu. Data-driven trajectory uncertainty quantification for climbing aircraft to improve ground-based trajectory prediction. *Anadolu Üniversitesi Bilim Ve Teknoloji Dergisi A-Uygulamalı Bilimler ve Mühendislik*, 18(2):323–345, 2017.

Charlie Vanaret, David Gianazza, Nicolas Durand, and Jean-Baptiste Gotteland. Benchmarking conflict resolution algorithms. In *International Conference on Research in Air Transportation (ICRAT), Berkeley, California, 22/05/12-25/05/12*, page (on line), http://www.icrat.org, may 2012. ICRAT. URL `http://www.icrat.org`.

Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

Vladimir N. Vapnik and Alexey Ya. Chervonenkis. The necessary and sufficient conditions for consistency of the method of empirical risk minimization. *Pattern Recogn. Image Anal.*, 1(3):284–305, 1991.