HAL

archives-ouvertes.fr

Ground-based prediction of aircraft climb : point-mass model vs regression methods

Richard Alligier, Mohammad Ghasemi Hamed, David Gianazza, Mathieu Serrurier

▶ To cite this version:

HAL Id: hal-00940961

https://hal-enac.archives-ouvertes.fr/hal-00940961

Submitted on 18 Mar 2014

Ground-based prediction of aircraft climb: point-mass model vs. regression methods

R. Alligier, M. Ghasemi Hamed, D. Gianazza
DSNA Toulouse, France
M. Serrurier
Institut de Recherche en Informatique de Toulouse, France

Introduction

Predicting aircraft trajectories with great accuracy is central to most operational concepts ([1], [2]) and automated tools that are expected to improve the air traffic management (ATM) in the near future.

On-board flight management systems predict the aircraft trajectory using a point-mass model describing the forces applied to the center of gravity. This 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 in this interval. The point-mass model requires knowledge of the aircraft state (mass, thrust, etc), atmospheric conditions (wind, temperature), and aircraft intent (target speed or climb rate, for example).

Many of these informations are not available to ground-based systems, and those that are available are not known with great accuracy. The actual aircraft mass is currently not transmitted to the ATM ground systems, although this is being discussed in the EUROCAE group in charge of elaborating the next standards for air-ground datalinks. The atmospheric conditions are estimated through meteorological models. Finally, the current ground-based trajectory predictors make fairly basic assumptions on the aircraft intent (see the "airlines procedures" that go with the BADA[1] model). These default "airline procedures" may not reflect the reality, where the target speeds are chosen by the pilots according to a cost index that is a ratio between the cost of operation and the fuel cost. These costs are specific to each airline operator, and not available in the public domain.

As a consequence, ground-based trajectory prediction is currently fairly inaccurate, compared to the on-board prediction. A simple solution would be to downlink the on-board prediction to the ground systems. However, this

[1]BADA: Base of Aircraft Data

is not sufficient for all applications: some algorithms (citation AlliotDurand) require the computation of a multitude of alternate trajectories that could not be computed and downlinked fast enough by the on-board predictor. There is a need to compute trajectory predictions in ground systems, for all traffic in a given airspace, with enough speed and accuracy to allow a safe and efficient 4D-trajectory conflict detection and resolution.

In this paper, we compare different ways to address this trajectory prediction problem, focusing on the aircraft climb with a 10 minutes look-ahead time. As a first approach, the point-mass model is tried with different settings for the model parameters, considering a constant CAS/Mach climb procedure where the aircraft first climbs at a constant Calibrated Air Speed (CAS) until it reaches the CAS/Mach crossover altitude and continues the climb at a constant Mach number. In this approach, the basic parameter setting consists in using the standard CAS and Mach values of the BADA climb procedures file, and a standard reduced thrust during climb, with an average reference aircraft mass. The second setting still uses the reference mass and standard thrust reduction factor, but the actual CAS is computed from the past aircraft positions.

The second approach is radically different and is based on regression methods. The predicted aircraft position is considered as a function $f(x, \theta)$ where x is a vector of of input variables and θ a vector of parameters. In our case, the input variables are the past aircraft positions, the observed CAS at the current altitude, the deviation of the air temperature from the standard atmosphere, and the predicted wind at different flight levels. The parameters (vector θ) must be adjusted using historical data so that the computed output fits the observed position as best as possible. Three regression methods are tried. In the first one, the function f is an artifial neural network predicting a couple (d, z) (along-track distance, altitude). In the second method, genetic programming is

used to find a function f predicting the altitude z: an initial population of functions follows a darwinian-like evolution during several generations, with crossovers, mutations, and selection of the best predictors at each generation. The third method uses fuzzy regression to predict a possibility function describing the uncertainties on the future aircraft position, instead of a single position or altitude as in the two other regression methods.

The rest of this paper is organized as follows: ...
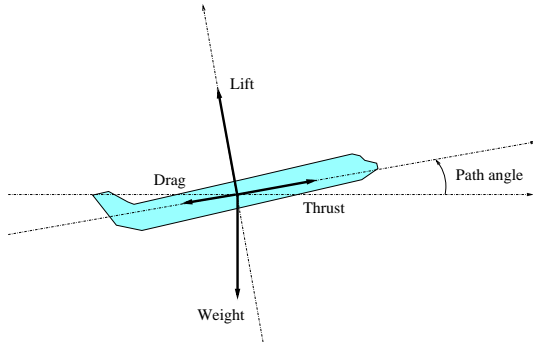
I. THE POINT-MASS MODEL

Figure 1.   Simplified point-mass model.

A. Simplified equations

Most ground systems use a simplified point-mass model, sometimes called *total energy model*, to predict aircraft trajectories. This model, illustrated on figure 1, describes the forces applying to the center of gravity of the aircraft and their influence the aircraft acceleration, making several simplifying assumptions[2]. It is assumed that the thrust and drag vectors are colinear to the airspeed vector, and that the lift is perpendicular to these vectors. Thus, projecting the forces on the airspeed vector axis, the longitudinal acceleration $a = \frac{dV_{\text{TAS}}}{dt}$ along the true airspeed (V_{TAS}) axis can be expressed as follows:

$$m.a = T - D - m.g.sin(\gamma) \qquad (1)$$

where T is the total thrust, D the aerodynamic drag of the airframe, m the aircraft mass, g the gravitational acceleration, and γ the path angle (i.e. the angle between the airspeed vector and the horizontal plane tangent to the earth surface).

[2]Note that more complex point-mass models have been proposed for UAV or fighter airplanes (see [3]), modeling also the side-slip angle.

Introducing the rate of climb/descent $\frac{dh}{dt} = V_{\text{TAS}}.sin(\gamma)$, where h is the altitude in meter, this equation can be rewritten as follows (see [4]):

$$(T - D).V_{\text{TAS}} = m.V_{\text{TAS}}.\frac{dV_{\text{TAS}}}{dt} + m.g.\frac{dh}{dt} \qquad (2)$$

Sereral equivalent forms of this equation can be used (see Eurocontrol BADA[3] User Manual), depending on what unknown variable is being calculated from the other known variables.

Actually using equation 2 to predict a trajectory requires a model of the aerodynamic drag for any airframe flying at a given speed through the air. In addition, we may need the maximum thrust, which depends on what engines the aircraft is equipped with. In the experiments presented here, the Eurocontrol BADA model was used to that purpose.

In addition, one cannot use equation 2 without prior knowledge of the initial state (mass, position, speed,...) of the aircraft, and also of the pilot's intents as to how the aircraft will be operated in the future (thrust law, speed law, or rate of climb). When the aircraft is operated at a given calibrated air speed (CAS[4]) or mach number, computing the true air speed (TAS) requires knowledge of the atmospheric conditions (the air temperature and pressure). Finally, as we need to predict the trajectory over the ground surface, and not only through the air, the wind intensity and direction are also required.

B. Aircraft operation during climb

Generally, when no external constraints apply during the climb, the aircraft is operated at constant CAS[5] and variable Mach number, until a specified Mach number is reached. Above this CAS/Mach crossover altitude, the aircraft is operated at a constant Mach number, and variable CAS. External constraints may apply, however. After take-off, the aicraft cannot exceed a specified maximum CAS until Flight Level 100[6] is reached. This first climb segment is followed by an acceleration at FL100, and then a second climb segment at a higher calibrated air-speed, until the CAS/Mach crossover altitude is reached.

In this paper, we shall consider only this second climb segment at constant CAS, followed by the constant Mach climb, as we are mostly interested in predicting the

[3]BADA: Base of Aircraft DAta
[4]CAS: Calibrated Air Speed, which can be assimilated to the speed indicated on the pilot's intruments.
[5]CAS: Calibrated Air Speed.
[6]FL100 = 10000 feet above isobar 1013 hPa.

aircraft trajectory in the en-route airspace. Note that some other air traffic control constraints may apply, that modify the aircraft operation during climb. For instance, the aircraft may be operated at a chosen rate of climb, on some flight segments, in order to be above a specified flight level over a given waypoint.

Even without such constraints, and assuming a climb at constant CAS/Mach, predicting the aircraft trajectory is not easy for ground systems. The actual CAS and Mach values are chosen by the airlines' operators, according to a cost index specific to each airline. The cost index, and the chosen CAS and Mach values, are not known to air traffic control systems today, although some studies show the improvements that such knowledge would provide in the trajectory prediction ([5], [6]).

II. REGRESSION METHODS

Regression methods aim at predicting an output y as a function of a given input x and a vector of parameters θ:

$$y = f(x, \theta) \tag{3}$$

In our trajectory prediction problem, we shall predict one of the following outputs, depending on the chosen method:

- a couple $(d(t), z(t))$ where d is the horizontal distance flown by the aircraft and z its altitude at time $t > t_0$, where t_0 is the current time,
- only the altitude $z(t)$,
- or possibility distributions $\Pi_d(t)$ and $\Pi_z(t)$ characterizing the uncertainties on the values of d and z at time t. These possibility distributions can be approximated by two quadruples $\Pi_d(t) = (d^1(t), d^2(t), d^3(t), d^4(t))$ and $\Pi_z(t) = (z^1(t), z^2(t), z^3(t), z^4(t))$ as illustrated on figure (* FIGURE TBD *).

The input x shall be a vector of values extracted from the following values:

- the current and previous aircraft states, characterized by $z[k]$, $d[k]$, *CAS*$[k]$, *Mach*$[k]$, with $k \in [-10, 0]$. The past trajectory is sampled every δt seconds. $z[k]$ denotes the value measured for the altitude z at time $t = t_0 + k.\delta t$. With this notation, $z[0] = z(t_0)$ is the current altitude, $z[-1]$ is the altitude δt seconds before t_0, and so on. The same notation applies for d, *CAS* and *Mach*,
- the difference between the actual air temperature at sea level and the air temperature of the International Standard Atmosphere (ISA) at sea level,

- the along-track and cross-track wind w and the temperature T at different altitudes.

The parameters θ must be adjusted using historical data, so that the computed outputs are as close as possible to the observed data. The performance of the tuned model shall be measured by assessing how the model generalizes on fresh inputs. k-fold cross validation can be used for that purpose.

In order to start with a relatively simple problem, we shall predict only one point (or one couple of fuzzy sets) of the future trajectory, N steps ahead. Let us now shortly describe the regression methods that are used to predict this future aircraft position (or uncertainty intervals).

III. REGRESSION USING NEURAL NETWORKS (NN)

Artificial neural networks are algorithms inspired from the biological neurons and synaptic links. An artificial neural network is a graph, with vertices (neurons, or units) and edges (connections) between vertices. There are many types of such networks, associated to a wide range of applications. Beyond the similarities with the biological model, an artificial neural network may be viewed as a statistical processor, making probabilistic assumptions about data ([7]). The reader can refer to [8] and [9] for an extensive presentation of neural networks for pattern recognition.

In our experiments, We used a specific class of neural networks, referred to as feed-forward networks, or multi-layer perceptrons (MLP). In such networks, the units (neurons) are arranged in layers, so that all units in successive layers are fully connected. Multi-layers perceptrons have one *input layer*, one or several *hidden layers*, and an *output layer*.

For a network with one hidden layer, the output vector $y = (y_1, ..., y_k, ..., y_q)^T$ is expressed as a function of the input vector $x = (x_1, ..., x_i, ..., x_p)^T$ as follows:

$$y_k = \Psi(\sum_{j=1}^{q} \theta_{jk}\Phi(\sum_{i=1}^{p} \theta_{ij}x_i + \theta_{0j}) + \theta_{0k}) \tag{4}$$

where the θ_{ij} and θ_{jk} are weights assigned to the connections between the input layer and the hidden layer, and between the hidden layer and the output layer, respectively, and where θ_{0j} and θ_{0k} are biases (or threshold values in the activation of a unit). Φ is an *activation function*, applied to the weighted output of the preceding layer (in that case, the input layer), and Ψ is a function applied, by each output unit, to the weighted sum of the activations of the hidden layer. This

expression can be generalized to networks with several hidden layers.

The output error – *i.e.* the difference between the desired output (target values) and the output y computed by the network – will depend on the parameters θ (weights and biases), that must be tuned by training the network, so as to minimize a chosen function of the output error. In our case, the minimized function is the sum of quadratic errors. The optimization method is either a gradient descent with momentum, or a BFGS quasi-Newton method. The activation function is the logistic sigmoãŕd, and the output function is the identity.

Neural network have already been applied to trajectory prediction, in [10]. However, they were used to predict both the climb and cruise flight segments, given a requested flight level, and lateral navigation as well. This approach, where the altitude error is likely to be small after the cruise flight level has been captured, is difficult to compare to our approach focused on minimizing the prediction error on the climb segment only. A mix of neural networks proved efficient for the chosen purpose, though.

IV. REGRESSION USING GENETIC PROGRAMMING (GP)

Genetic Programming (GP) is part of the Evolution Algorithm family. It is a population-based algorithm as can be seen on figure 2. Each individuals of the population encodes a computer program. A *fitness criterion* is assigned to those computer programs according their capacity in doing a predifined task. These fitnesses are then used for selecting individuals to recombined, giving then a new population. This approach is inspired from biology principles like genetic recombination and Darwinian selection principle.

In GP as it was popularized in [11], individuals are represented by a tree. They have a variable length. Typically, leafs contain inputs from the *terminal set T*, and nodes contain functions from the *function set F* which will be applied to the evaluation of its children. [11] has also defined a standard crossover and a standard mutation operation.

In a standard crossover, one subtree is randomly selected on each parents, and then the two selected subtree are then exchanged, as illustrated in figure 3.

In a standard mutation, one subtree is randomly selected and is then replaced by a generated tree, as showed in figure 4.

These recombination operators implies the *closure* property. Each functions shall accept any elements in
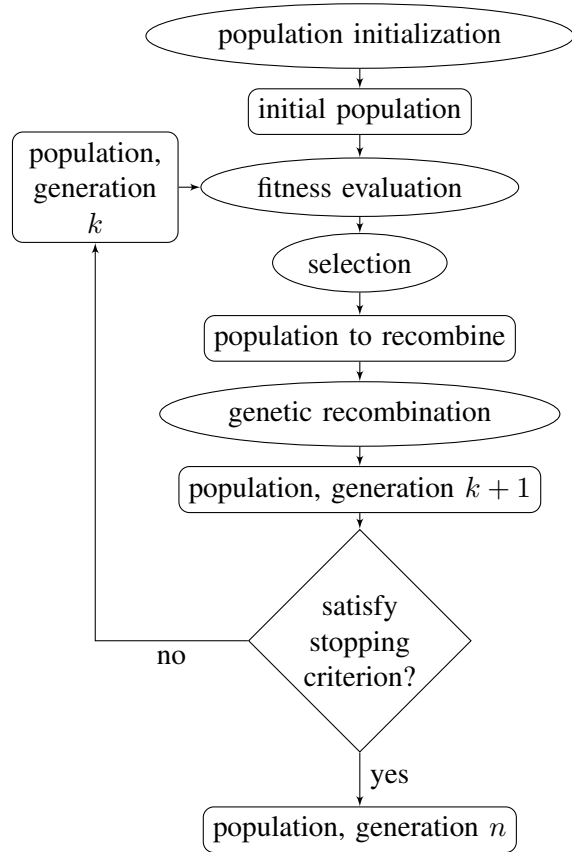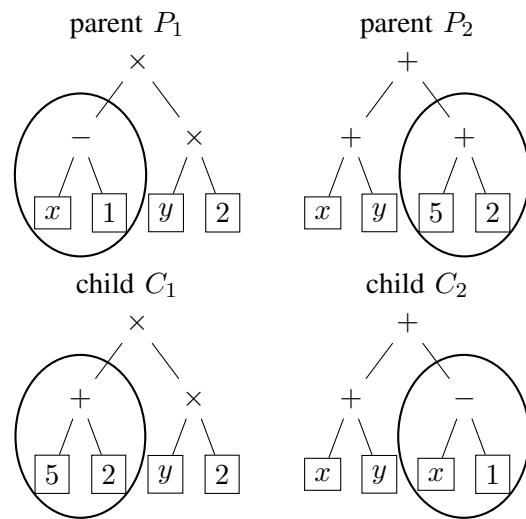

Figure 2.   Flow chart of Evolutionary Algorithms.

Figure 3.   Standard crossover.
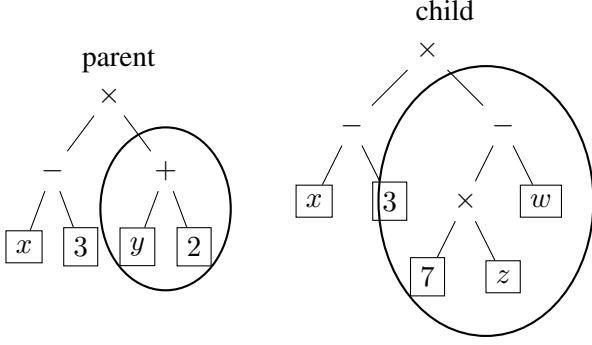
Figure 4. Standard mutation.

T and any output of the functions in F.

1) Our terminal set T and function set F: We have to find a function $f(x, \theta)$ parametrized by θ which predicts a variable y from a vector of input variables $x = (x_1, \ldots, x_n)$. A natural way to do that is to take $T = \{x_1, \ldots, x_n, \Re\}$ and $F = \{+, *, -, /, cos, sin\}$ for instance, with $\Re$ a variable which is replaced by a randomly generated constant at the creation of the tree. In fact, this method to generate constant terms in our expression is said to be inefficient. In order to cope with this issue, we have hybridized GP with a multiple regression technique, the Ridge regression [12]. The execution of a tree gives us a set of functions $G = \{g_1, \ldots, g_k\}$ which will be used to build a function $f(x, \{\lambda_0, \ldots, \lambda_k\}) = \lambda_0 + \sum_{i=1}^{k} \lambda_i g_i(x)$ with its parameters adjusted by a Ridge regression. The g_i generated are piecewise monomials. We have defined $T = \{\{x_1\}, \ldots, \{x_p\}\}$. If we consider n input variables $x_1, \ldots, x_n$, N instances in our training set and two sets $A = \{a_1, \ldots, a_{k_A}\}$ and $B = \{b_1, \ldots, b_{k_B}\}$, we have build operators to combine these two sets of functions A and B:

- $\oplus(A, B) = A \cup B$
- $\otimes(A, B) = \bigcup_{i=1}^{k_A} \{a_i.b_1, \ldots, a_i.b_{k_B}\}$
- $\Delta_{i,p}(A, B) =$
  $\{\delta(\{x_i \leq v_{i,p}\}, a_1), \ldots, \delta(\{x_i \leq v_{i,p}\}, a_{k_A})\} \cup$
  $\{\delta(\{x_i > v_{i,p}\}, b_1), \ldots, \delta(\{x_i > v_{i,p}\}, b_{k_B})\}$
  with :
  - $v_{i,p}$, the p-th bigger value of x_i observed in our training set.
  - $\delta(COND, expr)(x) =$
    $\begin{cases} expr(x) & \text{if } \forall \ cond \in \ COND, \ cond(x) \\ 0 & \text{otherwise} \end{cases}$
    where x is an instance.

With these definitions, we can notice that:

- $\delta(COND_1, expr_1).\delta(COND_2, expr_2) = \delta(COND_1 \cup COND_2, expr_1.expr_2)$
- $\delta(COND_1, \delta(COND_2, expr_2)) = \delta(COND_1 \cup COND_2, expr_2)$
- $COND$ defines a multi-dimensionnal interval though it can be simplified. For instance $\{8 < x_1, 9 < x_1, x_1 \leq 14, x_1 \leq 15\}$ can be simplified in $\{9 < x_1, x_1 \leq 14\}$. Thus, no more than two conditions have to be associated with one variable x_i, making conditions sets easy to compare between each other.

In our implementation we have chosen $F = \{\oplus, \otimes\} \bigcup \bigcup_{i=1}^{n} \bigcup_{p=1}^{N-1} \{\Delta_{i,p}\}$ as the function set. Consequently, we can assume that the set encoded by a tree only contains expressions such as $\prod_{i=1}^{n} x_i^{k_i}$ or $\delta(COND, \prod_{i=1}^{n} x_i^{k_i})$ with $k_i \in \mathbb{N}$. Thus, expressions are easy to compare, making the $\cup$ operation manageable on our function sets.

With this choice, $f(x, \{\lambda_0, \ldots, \lambda_k\})$ is a piecewise polynomial. Conditions are used there as we assume that the trajectory prediction problem to be piecewise.

2) Fitness criterion: We have to assess a *fitness* to each candidate function $f(x, \{\lambda_0, \ldots, \lambda_k\})$. We want this function to make good prediction on unseen data. Typically, we learn parameters $\lambda_0, \ldots, \lambda_k$ from a finite training set containing N instances. Using those N instances, the more we have parameters, the less their estimation will be reliable. In order to take into account this issue, we have used the AIC_c criterion [13], [14] defined as $AIC_c = N.log(MSE) + 2(k+2) + \frac{2(k+2)(k+3)}{N-k-2}$ with $MSE = \min_{\lambda_0, \ldots, \lambda_k \in \mathbb{R}} \frac{1}{N} \sum_{i=1}^{N} (y^{(i)} - f(x^{(i)}, \{\lambda_0, \ldots, \lambda_k\}))^2$.

The algorithm was parametrized as follow :

$T = ROCD + CAS + DCAS + MACH + DMACH + ESF + Dtemp0_ISA + W_along_FL190 + W_cross_FL190 + Temp_FL190 + W_along_FL250 + W_cross_FL250 + Temp_FL250 + W_along_FL310 + W_cross_FL310 + Temp_FL310 + W_along_FL340 + W_cross_FL340 + Temp_FL340 + W_along_FL370 + W_cross_FL370 + Temp_FL370 + DdAIR_minus9 + DdAIR_minus7 + Dz_minus7 + DdAIR_minus5 + Dz_minus5 + DdAIR_minus3 + Dz_minus3 + DdAIR_minus1 + Dz_minus1$

General parameters	
population size	200
number of generation	100
maximum depth	5
Initialization parameters	
initialization method	*ramped half-and-half*
$P(\{\oplus, \otimes\}\|F)$	0.91
Fitness	
α_{ridge}	0.001
Scaling	*sigma scaling* [15]
Selection	
Selection method	*stochastic remainder without replacement selection* [15]
Recombination	
$P_{crossover}$	0.5
$P_{mutation}$	0.45

V. Interval regression using K nearest neighbours (I-KNN)

A. Possibility theory

Possibility theory, introduced Zadeh [16], [17], was initially created in order to deal with imprecision and uncertainty due to incomplete information. This kind of uncertainty may not be handled by probability theory, especially when a priori knowledge about the nature of the probability distribution is lacking. A possibility distribution π is a function from Ω, the universe of discourse ($\mathbb{R}$ in our case), to ($\mathbb{R} \rightarrow [0,1]$). The definition of the possibility measure Π is based on the the possibility distribution π such that :

$$\Pi(A) = sup(\pi(x), \forall x \in A). \qquad (5)$$

The α-cut of a possibility distribution A_α is the interval for which all the points located inside, have a possibility membership $\pi(x)$ greater or equal to α. We have :

$$A_\alpha = \{x | \pi(x) \geq \alpha, x \in \Omega\}. \qquad (6)$$

One interpretation of possibility theory is to consider a possibility distribution as a family of probability distributions (see [18] for an overview). Thus, a possibility distribution π will represent the family of the probability distributions Θ for which the measure of each subset of Ω will be bounded its possibility measures :

$$\Theta = \{P | \forall A \in \Omega, P(A) \leq \Pi(A)\}. \qquad (7)$$

Given a probability distribution p, a confidence interval I_α is a subset of Ω such as $P(I_\alpha) = \alpha$. We define I_α^*, also referred as quantile, as the smallest confidence
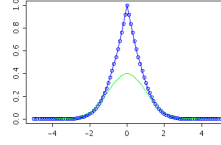

Figure 5. The maximum specified possibility distribution for N(0,1).

interval with probability measure equal to α. Thus, an alternative of the equation 7 is:

$$\forall \alpha \in [0,1], p \in \Theta, I_{p,(1-\alpha)}^* \subseteq A_\alpha. \qquad (8)$$

Thus, a possibility distribution encode a family of distributions for which each quantile is bounded by α-cuts. In many cases it is desirable to move from the probability framework to the possibility framework. Probability-possibility transformation [19] is based on the maximum specificity principle. Given a probability distribution p, what we obtain is the possibility distribution π^* which is the most specific one that bound p. Given p we compute π^* as follows :

$$\forall x \in \Omega, pi^*(x) = max_{\alpha, x \in I_\alpha}(1 - \alpha). \qquad (9)$$

Then, in the spirit of equation 8, given p and its transformation π^* we have :

$$A_{1-\alpha}^* = I_\alpha^*$$

Figure 5 presents the maximal specific transformation of a normal distribution p is the π possibility distribution.

B. Possibility distribution for a Gaussian family computed on a set of data

When we suppose that our data comes from a Normal distribution with unknown mean (μ) and variance (σ^2). The solution that comes immediately to our mind is to estimate μ and σ from the data. However, computing a value of these parameters that is statistically faithful requires a lot of data. One solution is to compute confidence intervals confidence interval for the mean and the variance. Based of the χ^2 distribution, the confidence intervals $[\mu_{min}, \mu_{max}]$ and $[\sigma_{min}, \sigma_{max}]$ will contains respectively 95% of the possible values for μ and σ given a set of data. We propose to compute the maximal specific possibility distribution that contains all the normal distributions that have parameters in the confidence intervals ($\Theta = \{g | g = N(\mu, \sigma^2), \mu \in [\mu_{min}, \mu_{max}], \sigma^2 \in [\sigma_{min}^2, \sigma_{max}^2]\}$). We compute le possibility distribution π_Θ^* as follows :

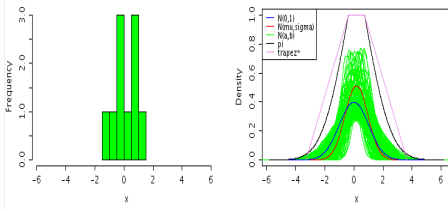- $\forall x \in [\mu_{min}, \mu_{max}] : \pi_\Theta^*(x) = 1$

Figure 6. An example of the possibility distribution for the family Θ, with a confidence level of 0.95 and a dataset with n=10.

- $\forall x < \mu_{min} : \pi_\Theta^*(x) = sup(1 - P(I_\alpha^*), x \in I_\alpha^*)$ with $p = N(\mu_{min}, \sigma_{max}^2)$
- $\forall x > \mu_{min} : \pi_\Theta^*(x) = sup(1 - P(I_\alpha^*), x \in I_\alpha^*)$ with $p = N(\mu_{max}, \sigma_{max}^2)$

where G is the probability measure associated with the density g. In the following, in a simplification purpose, we will call respectively, the normal distribution built from the estimated mean and the estimated variance and the maximum specified possibility distribution encoding the family Θ described above, $N(\mu_e, \sigma_e)$ and π_Θ^*. Figure 6 present the distribution π_Θ^* for a data set of size 10 that follows a Gaussian distribution. In this figure the Θ family is represented by a set of green distribution functions $N(a, b)$, the distribution π_Θ^* is in black. When the dataset size grows, π_Θ^* converges to the normal distribution. Thus, the size of π_Θ^* (his specificity) depends on the quantity of data available and allows us to have a statistically significant representation of the data.

C. K-nearest neighbours (KNN)

The k-nearest neighbours(KNN) algorithm is a regression method which finds the response value of the input data based on closest training instances. For any given input instance X, the KNN algorithm finds the K nearest instances ($Kset_X$) and chooses the mean of their response values (Y) as the estimated value ($Y_{estimated}$). In our case, we are not only interested in obtaining the most probable position of the plane, but we are also interested to find the interval which contains the response value with an high probability. With global regression approaches, this interval can be computed by using the standard deviation of the error (RMSE). Since it is a global estimation, the size of the interval is constant, so it does not depend on the inputs. Moreover, it supposes that the distribution of the error follows an a priori known distribution (usually a Gaussian one) and that we have enough data to estimate precisely the parameters. The advantage of the KNN algorithm is that it can give a

local estimation of the data distribution. However, the imprecision about the parameters of the function depends on the value of k. When k is high the parameters are precise, but the distribution will contain a lot of data that are different to the input one. Then, quantile will be very large. On the contrary, if k is low, the k-set will contain data that are close to the input but the imprecision about the parameters of the distribution will be high. In our method, we propose, given k, to compute the possibility distribution that bounds all the Gaussian distributions that may have generated the k-set (see previous section). For each input value we choose the value of k that is the best trade off between the precision and the size of the quantile. The algorithm is described in Alg The idea

---

Algorithm 1 For any input X, finds the minimum interval which should contain the response value

---
1: $Kset_X \leftarrow$ Find the K nearest neighbours of X
2: $MIN_K \leftarrow K$
3: $IntervalSize_{min} \leftarrow Inf$
4: **for all** $i \in 5, \ldots, K$ **do**
5:     Compute $[\mu_{min}, \mu_{max}]$ and $[\sigma_{min}, \sigma_{max}]$ w.r.t. the i-th first example in $Kset_X$
6:     $IntervalSize \leftarrow G^{-1}(0.975, \mu_{max}, \sigma_{max}^2) - (G^{-1}(0.25, \mu_{min}, \sigma_{max}^2)$
7:     **if** $IntervalSize \leq PIIntervalSize_{min}$ **then**
8:         $MIN_K \leftarrow i$
9:         $IntervalSize_{min} \leftarrow IntervalSize$
10:    **end if**
11: **end for**

---

of the approach is to have intervals that guarantee to contain the expect quality of data. For instance, given k, the 0.05-cut of π_Θ^* will contain at least 95% of the data since it contains the 0.95 quantiles of all the Gaussian distribution that may have generated the k-set. This approach may be useful for instance for detecting conflict or for defining safe area zone because it will take into account all the cases that are statistically possible.

VI. DATA AND EXPERIMENTAL SETUP

A. Data pre-processing

Recorded radar tracks from Paris Air Traffic Control Center were used to build the patterns used in the regression methods. This raw data is made of one position report every 1 to 3 seconds, over two months (july 2006, and january 2007). In addition, the wind and temperature data from Meteo France are available at various isobar altitudes over the same two months.

The raw Mode C altitude[7] has a granularity of 100 feet. So the recorded aircraft trajectories were smoothed, using a local quadratic model, in order to obtain: the aircraft position (X,Y in a projection plan, or latitude and longitude in WGS84), the ground velocity vector (V_x, V_y), the smoothed altitude (z, in feet above isobar 1013.25 hPa), the rate of climb or descent (ROCD). The wind (W_x, W_y) and temperature (T) at every trajectory point were interpolated from the meteo datagrid. The temperature at isobar 1000 hPa was also extracted for each point, in order to compute a close approximation of $(\Delta T_0)_{\text{ISA}}$, the difference between the actual temperature and the ISA model temperature at isobar 1013.25 hPa (mean sea level in the ISA atmospheric model). This $(\Delta T_0)_{\text{ISA}}$ is one of the key parameters in the BADA model equations.

Using the position, velocity and wind data, we computed the true air speed (TAS), the distance flown in the air (dAIR), the drift angle, the along-track and cross-track winds (W_{along} and W_{cross}). The successive velocity vectors allowed us to compute the trajectory curvature at each point. The actual aircraft bank angle was then derived from true airspeed and the curvature of the air trajectory. The climb, cruise, and descent segments were identified, using triggers on the rate of climb or descent to detect the transitions between two segments.

Finally, the BADA model equations were used to compute additional data, such as: calibrated airspeed (CAS), Mach number (M), energy share factor[8] (ESF), as well as the derivatives of these quantities with respect to time.

B. Filtering and sampling climb segments

As our aim is to compare several prediction models, we focused on a single aircraft type (Airbus A320), and selected all flights of this type departing from Paris Orly (LFPO) or Paris Roissy-Charles de Gaulle (LFPG). The trajectories were then filtered so as to keep only the climb segments. An additionnal 40 seconds were clipped from the beginning and end of each segment, so as to remove climb/cruise or cruise/climb transitions.

The trajectories were then sampled every 15 seconds, with time and distance origins at the point P_0 where the climb segment crosses flight level FL180[9]. The trajectory

segments were sampled so as to obtain 10 points preceding P_0, and a number of points after P_0, depending on the chosen look-ahead time. So the trajectory observed during the preceding time steps (2 minutes 30 seconds), can be used to predict the aircraft position at one or several future time steps. The predicted position can be compared to the actual aircraft position at the same time step.

Trajectories exhibiting a bank angle greater than 5 degrees were discarded, so that the influence of trajectory turns on the rate of climb can be neglected. This allows us to disregard the lateral navigation in our trajectory prediction problem, and focus on the longitudinal and vertical dimensions of the trajectory.

C. Building patterns for regression

The regression models $y = f(x, \theta)$ are tuned and assessed using sets of patterns (x, d), where x is an input vector, and d is the corresponding desired output that can be compared to the computed output y. These patterns, that we have already described in section II were extracted from the sampled climb segments. 1500 patterns were randomly chosen, to build the set used in our experiments.

Each pattern used for regression contains the current ground speed, true and calibrated air speed, Mach number, and their derivatives with respect to time, the energy share factor, the altitude variations and distance flown for the ten preceding time steps, and also the predicted wind and temperature at several altitudes that the aircraft may cross in the look-ahead time. It also contains the target variables: distance flown, in the air or above the ground, and altitude reached after N time steps in the future.

D. Principal component analysis

The final patterns set contains 79 numerical variables, measured for 1500 aircraft climbs. There are 76 explanatory variables, and 3 variables to explain. A principal component analysis was performed on the explanatory variables, so as to reduce the dimensonality and avoid redundant input variables in the trajectory prediction.

Figure 7 shows the standard deviations of the principal components: 9 components have a standard deviation above 1, and 7 other components are between 0.5 and 1.

Principal components are linear combinations of the initial variables, that we can use as explanatory variables in the regression method. This reduces the dimensonality to 10 to 15 significant principal components, instead of the 76 initial variables. One must keep aware, however, that using linear combinations representing projections

---

[7]This altitude is directly derived from the air pressure measured by the aircraft. It is the height in feet above isobar 1013.25 hPa.

[8]The energy share factor (ESF) says how much of the energy is devoted to climb or to longitudinal acceleration.

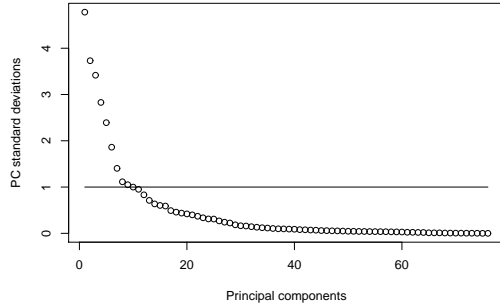[9]FL180: 18000 feet above isobar 1013 hPa.

Figure 7.   Principal components standard deviations.

on a basis of orthogonal vectors may not take account of some non-linearities in the initial variables.

E. Experimental setup

The methods applied to our prediction problem for climbing aircraft are listed in table I.

BADA	BADA point-mass model, using the reference mass for each aircraft and BADA values for the constant CAS/Mach, assuming reduced climb (eq. 3.8.1 and 3.8.2, p.22 in [4]), and taking account of the $(\Delta T_0)_{\text{ISA}}$ temperature difference.
BADA(obs)	Same BADA model as above, but using the CAS observed at t_0, and the BADA target Mach number
LR	Ordinary least squares linear regression
NN	Regression with neural networks
GP	Regression with genetic programming, using Akaike's AIC to assess the fitness of each prediction function in the population.
I-KNN	Interval regression using K-nearest neighbours, returning intervals of uncertainty

Table I
METHODS APPLIED TO ALTITUDE PREDICTION

In our experiments, these methods are scored using a 10-fold cross-validation on the dataset described in section VI-C. This set is split in ten subsets. Nine of these subsets are used to tune the model parameters θ, and the remaining subset is used to assess the model performance. This operation is repeated 10 times, cycling through the subsets. The models' performace is assessed over the ten runs, considering the mean score, the standard deviation, and also confidence intervals for the computed output.

Uncertainty on the prediction can be assessed in several ways. Once the model parameters have been tuned on the training set (i.e. the concatenation of 9 subsets

used in cross-validation), we can compute a theoritical 95%-confidence interval using the root mean square error (RMSE) observed on this training set, assuming a Gaussian distribution of the error in altitude (and also in distance, if predicted). We can then count how many instances of the validation subset actually fall within this confidence interval.

We can also assess the uncertainty on the predicted altitude (or distance) by considering the actual distribution of errors in the training set. We can then assume a Gaussian distribution, but with some uncertainty on the mean value and standard deviation. This gives us a family of Gaussian distributions from which we can compute a possibility distribution (see section V-B). This is the approach chosen in the possibilistic regression using K-nearest neighbours. For every instance in the validation set, the K nearest instances of the "training set"[10] are found, using a distance between vectors of variables. The possibility distribution is computed on these K neighbours. In theory, the 0.05-cut of this possibility distribution is guaranteed to contain at least 95% of the data. We can then check if our instance in the validation set falls within this interval or not. This is repeated for all instances in the validation set, such providing the ratio of predictions actually within the confidence interval of variable size (there is an interval for each instance).

VII. RESULTS

Method	MAE	RMSE
BADA	1440 (79)	1824 (95)
BADA(obs)	1440 (77)	1819 (86)
LR	745 (35)	965 (47)
NN	841 (47)	1080 (55)
GP	744 (28)	964 (41)
I-KNN	-	-

Table II
AVERAGE PREDICTION ERRORS (AND STANDARD DEVIATIONS) ON THE ALTITUDE (IN FEET) FOR AIRBUS A320 AIRCRAFT, USING 15 PRINCIPAL COMPONENTS AS INPUT, WITH THE REFERENCE POINT AT FL180 AND A 10-MINUTES LOOK-AHEAD TIME.

Table II shows the prediction errors (mean absolute error, and root mean squared error) over the 10 runs of the cross-validation, for all tested methods. The 15 principal components of higher variance were used as input to the regression methods. This selection was made by

---

[10]This is not actually a training set, as there is no training when using KNN, but actually a dataset of examples.

prior trials, adding successively the principal components until no significant improvement was observed.

All regression methods perform significantly better than the BADA point-mass model. There are several factors explaining the poor performance of the point-mass models. The parameters'choice assumed a constant CAS/Mach climb at economic thrust, and the same reference mass for all aircraft, which is actually not the case in reality. Also, the regression method use the past trajectory to predict the future altitude, whereas our BADA models do not. Using the observed CAS instead of the BADA standard CAS does not improve the results on altitude prediction.

We also observed that the mean predicted altitude with BADA models is biased toward lower altitudes. The mean error, averaged over the 10 runs, is -735 feet for the first BADA model, and -603 feet for the second BADA model. After some investigation, it appears that the CAS is highly sensitive to the air density, and that much more realistic values are obtained when using a more accurate temperature model (Meteo France data at successive altitudes) than when extrapolating from $(\Delta T_0)_{\text{ISA}}$. At the time of this publication, this was not implemented in our BADA model. It is expected that it would give errors more centered around zero, but with a similar dispersion than today's implementation.

It came as a surprise that non-linear regression methods did not perform better than the ordinary least squares linear regression. There may be several explanations to this. Using the principal components as inputs does favour linear methods. In addition, tuning the parameters with the ordinary least squares linear regression can be done with an exact method, whereas non-linear methods require iterative approximations, that may have difficulties to find the optimum when using very noisy data.

Method	Ratio in theoretical 95% interval	Theoretical 95% $|\delta z|$	Actual 95% $|\delta z|$
BADA	0.92 (0.025)	3279 (20)	3593 (249)
BADA(obs)	0.93 (0.021)	3369 (19)	3674 (260)
LR	0.94 (0.013)	1863 (10)	2015 (187)
NN	0.905 (0.030)	1846 (92)	2203 (215)
GP	0.93 (0.019)	1818 (26)	1923 (162)
I-KNN	0.986	3376 (714) min. 417 max. 6082	-

Table III
UNCERTAINTY ON THE ALTITUDE PREDICTION (AIRBUS A320), FOR A REFERENCE POINT AT FL180 AND A 10-MINUTES LOOK-AHEAD TIME.

Some results on the uncertainty of the altitude prediction are shown in Table III. The second column shows the ratio of predictions, computed with instances from the validation set, that actually fall within the 95% confidence interval computed using the training set. For all methods but I-KNN, this interval is simply the interquantile interval corresponding to the 95% confidence, assuming a Gaussian distribution of the error. For I-KNN, this is the 0.05-cut of the possibility distribution computed from a family of Gaussian distributions. The third column shows the width of this theoretical interval. As this interval is of variable size for I-KNN, the lower, mean, and higher half-width observed on all instances are given. The last column shows the actual value of $|\delta z|$ (the interval half-width) for which 95% of the observed altitudes z_obs fall in the interval $[z - \delta z, z + \delta z]$, where z is the predicted altitude.

CONCLUSION

In this paper, we have applied several methods to the prediction of altitude, focusing on a single aircraft type (A320). The aim was to compare these methods when predicting the altitude of climbing aircraft 10 minutes ahead, starting from an initial point at flight level FL180, and possibly using the past trajectory to improve the prediction. Radar and Meteo data recorded over two months (july 2006, january 2007) were used to build a dataset of explanatory and target variables. A principal component analysis of this data allowed us to reduce the dimensionality to 10 to 15 significant components, instead of the 76 initial explanatory variables. The models are compared by performing a 10-fold cross-validation on a set of 1500 climb segments.

Our results show that the regression methods perform better than the point-mass model. This is not surprising as the former learn from the observation of the past trajectory, whereas the point-mass model uses the same standard values for most parameters (mass, power reduction, target speeds) for all aircraft. The genetic programming approach proved efficient, although not better, on our noisy data, than standard regression methods.

Two different kinds of predictive methods have been applied to our problem, with two different ways to compute confidence intervals on the predicted altitude. In the first approach (LR, GP, NN, and also BADA models), the target variable to predict is the altitude itself. A probabilistic confidence interval can be obtained by considering the interquantile interval, assuming a Gaussian distribution of the error. This confidence interval, although statistically relevant when measured on an

infinity of instances, is not guaranteed to actually contain the expected ratio of accurate predictions (e.g. 95%) for a limited number of instances. This explains why we observed less than 95% of the predictions actually falling in the theoretical confidence interval, with these methods.

In the second approach, the interval regression using K-nearest neighbours (I-KNN) returns a confidence interval, instead of the altitude itself. This method still assumes a Gaussian distribution of the error, but with some uncertainty on the distribution's parameters. This gives a family of probability distributions, from which we can deduce a possibility distribution. I-KNN compute one such possibility distribution for each instance, using the instance's nearest neighbours in the examples' dataset. So the size of the confidence interval obtained from this distribution depends on the quantity of data in the neighborhood of the considered instance. Obviously, these intervals are generally larger than the probabilistic confidence intervals obtained with standard methods. However, they are statistically guaranteed to contain at least the expected ratio of accurate predictions. This is confirmed in our experiments, where we found 98.6% instances actually falling in the 95% confidence interval.

From an operational point of view, the proposed methods could be applied to the detection of potential conflicts between trajectories. Standard regression methods could be used to provide a relatively narrow probabilistic interval allowing us to detect conflicts with a great look-ahead time, although with some uncertainty. With smaller look-ahead times, when more certainty is required before actually deviating conflicting trajectories, interval regression could provide guaranteed confidence intervals on the altitude prediction.

In future works, we plan to conduct a more thorough analysis of the available data, trying to filter the climb segments according to the actual aircraft mode of operation (constant CAS or constant ROCD, for example), so as to obtain less noisy data. We could then learn the aircraft climbs in a specific operation mode, thus giving a better chance to the point-mass model. It could also be interesting to learn some of the point-mass model parameters (mass, thrust law) from the observed data. Concerning the regression methods, we could try to improve the genetic programming approach by introducing elements of the point-mass model in the population of predictors.

REFERENCES

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

[2] H. Swenson, R. Barhydt, and M. Landis. Next Generation Air Transportation System (NGATS) Air Traffic Management (ATM)-Airspace Project. Technical report, National Aeronautics and Space Administration, 2006.

[3] F. Imado T. Kinoshita. The application of an uav flight simulator - the development of a new point mass model for an aircraft. In *SICE-ICASE International Joint Conference Conference*, 2006.

[4] A. Nuic. User manual for base of aircarft data (bada) rev.3.7. Technical report, EUROCONTROL, 2009.

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

[6] Study of the acquisition of data from aircraft operators to aid trajectory prediction calculation. Technical report, EUROCONTROL Experimental Center, 1998.

[7] M. I. Jordan and C. Bishop. *Neural Networks*. CRC Press, 1997.

[8] C. M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1996. ISBN: 0-198-53864-2.

[9] B. D. Ripley. *Pattern recognition and neural networks*. Cambridge University Press, 1996. ISBN: 0-521-46086-7.

[10] Y. Le Fablec. *Prévision de trajectoires d'avions par réseaux de neurones*. PhD thesis, Institut National Polytechnique de Toulouse, 1999.

[11] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*. The MIT Press, December 1992.

[12] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[13] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716 – 723, dec 1974.

[14] Clifford M. Hurvich and Chih-Ling Tsai. Model selection for extended quasi-likelihood models in small samples. 51:1077–1084, 1995.

[15] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, 1 edition, January 1989.

[16] L.A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1(1):3–28, 1978.

[17] D. Dubois and H. Prade. *Fuzzy sets and systems - Theory and applications*. Academic press, New York, 1980.

[18] D. Didier. Possibility theory and statistical reasoning. *Computational Statistics and Data Analysis*, 51:47–69, 2006.

[19] Didier Dubois, Henri Prade, and Sandra Sandri. On possibility/probability transformations. In *Proceedings of Fourth IFSA Conference*, pages 103–112. Kluwer Academic Publ, 1993.