New helicopter model identification method based on flight test data

S. De Jesus Mota
R. M. Botez
Ruxandra.Botez@etsmtl.ca
Ecole de technologie supérieure
Montréal, Canada

ABSTRACT

A helicopter model has been identified and validated for flight conditions defined by altitudes, speeds, loadings and centre of gravity positions. To identify the helicopter models, 2-3-1-1 multistep control inputs were performed by the pilot to excite all helicopter modes. Then, each estimated signal has to remain in tolerance margins defined by the Federal Aviation Administration. Three methods were used to observe the system outputs from its states: a fuzzy logic method, a linear method optimised with a neural network algorithm and a classical method. Because of random effects when gathering data, classical method did not give good enough results. The fuzzy logic method was not robust enough so that output plots showed peaks that could be felt by the pilot. Then, because the model could be implemented in a simulator for the pilot training, the pilot feedback is very useful in order to compare the reality with the results of the mathematical model. When the outputs are obtained from the measured state variables, the linear method gave the best results.

NOMENCLATURE

CG	centre of gravity
CL	closed-loop
DOF	degree of freedom
FAA	Federal Aviation Administration
L/H/A/F	Light / Heavy / Aft / Forward
MISO	multiple input single output
MIMO	multiple input multiple output
OL	open-loop
a	parameter of the **A** matrix
A	state matrix
A_x, A_y, A_z	linear accelerations on the x, y and z axes
b	parameter of the **B** matrix
B	control matrix
coll	collective command position
C	output matrix
Corr	correlation coefficient
Cov	covariance
e	error vector
FIT	fit coefficient
$\dot{h}$	altitude rate
Id	identity matrix
J	cost function
lat	lateral cyclic command position
long	longitudinal cyclic command position
m	output number
n	state number
p	roll attitude rate
ped	pedals command position
q	pitch attitude rate
r	yaw attitude rate
u	longitudinal speed component in body axis system
u	input vector
U	Theil's coefficient

Paper No. 3564. Manuscript received 6 April 2010, revised version received 22 September 2010, 2nd revised version received 7 December 2010, accepted 3 February 2011.

v	lateral speed component in body axis system
Var	variance
w	vertical speed component in body axis system
$\mathbf{x}$	state vector
$\mathbf{y}$	output vector
φ	roll attitude angle
γ	forgetting factor
λ	lagrangian parameter
σ	standard deviation
θ	pitch attitude angle
ψ	yaw attitude angles
η	learning rate
μ	momentum

Superscripts

^	estimated
T	transpose

Subscripts

des	desired
in	input
new	new value
old	old value
out	output
real	measured

1.0 INTRODUCTION

System identification is a process that provides a model that best characterises (in some least-square sense) a system's measured responses to controls[1]. Parametric modelling requires a priori assumptions about the model such as its order, its degree of coupling, the structure of the equations of motion and the initial estimates of key parameters. In order to answer to these problems, a fully-coupled rigid body parametric model was studied, which has six degrees of freedom (DOF) with three translations (longitudinal, lateral and vertical) and three rotations (pitch, roll and yaw).

Two methods are usually used to identify a model: the time domain and the frequency domain[1]. A comparison of the frequency and time domain methods was given[11]. Frequency-domain identification used spectral methods to determine frequency responses between selected input and output pairs. Then, least-squares fitting techniques were used to obtain closed-form analytical transfer-function linear input-to-output models. Time-domain identification required the selection of a state-space model structure, which may be linear or nonlinear. Model parameters were identified by least-square fitting of the response time-histories or by maximum likelihood methods.

Various neural network methodologies were used until now for helicopter model identification. These methodologies are here described to show their differences with respect to the methodology given in this paper. An artificial neural network (ANN) based helicopter identification system was proposed[16]. The feature vectors were based on both the tonal and the broadband spectrum of the helicopter signal. ANN pattern classifiers were trained using various parametric spectral representation techniques. Specifically, linear prediction, reflection coefficients, and line spectral frequencies (LSF) were compared in terms of recognition accuracy and robustness against additive noise. Finally, an eight-helicopter ANN classifier was evaluated.

Neural Network Identification (NNID) for modelling the dynamics of a miniature Eagle helicopter was presented[17]. Off-line and on-line identification was carried out for both coupled and decoupled dynamics of the helicopter from the flight test data. Identification results and error statistics were provided. The off-line identification performed better due to sufficient training time and data. Results indicated that neural network based black – box methods would be suitable for modelling the nonlinear dynamics of the helicopter and could be further applied for the design of automatic flight control system (AFCS).

A set of optimisations was presented in learning algorithms commonly used for training radial basis function (RBF) neural networks[18]. These optimisations were applied to a RBF neural network used in identifying helicopter types, processing their rotor sounds. The first method used an optimum learning rate in each iteration of training process. This method increased the speed of learning process and also achieved an absolute stability in network response. Another modification was applied to quick propagation (QP) method as a generalisation that attained more learning speeds. Finally, the general optimum steepest descent (GOSD) method was used, which contained both improvements in learning RBF networks. All modified methods were employed in training a system that recognised helicopters' rotor sounds exploiting a RBF neural network.

A study of a novel acoustic helicopter identification system was described[19]. The system consisted of a pattern classifier that learned to distinguish helicopter types based on the relationship between the spectral amplitudes of the main rotor fundamental frequency and its first seven harmonics. Two pattern classifiers were evaluated; a statistically based Bayes classifier, and an adaptive neural network classifier. Measured noise of three helicopters was used to train and test the classifiers. After an initial calibration or training process, the Bayes classifier correctly identified the helicopters 67% of the time. The neural network classifier was correct 65% of the time. These results demonstrated that the spectral amplitudes of the main rotor tones can be used for helicopter identification if another source of information was also available in the identification. The paper described statistical algorithms for pattern classification, the neural network, and the measured helicopter noise used to evaluate the classifiers.

In this paper, two major problems are solved. The first deals with determination of the dynamic behaviour of the helicopter by a linear model. In the second problem, three methods were compared with the aim to observe the system output from the measured (open-loop study) and the estimated states (closed-loop study) of the system. The parameters used as states are the linear velocities, the angular rates and the Euler angles, while the observed parameters are the linear accelerations.

The flight test data and the data preprocessing method are presented first. Next, the theories used to determine the state equation in open-loop and the optimisation procedure required to obtain the estimated signals in closed-loop are detailed. Principles of neural network and fuzzy logic methods described in the literature9 were used in the identification of the rotorcraft model based on flight tests. All results are found to be in accordance with the Federal Aviation Administration (FAA) rules. In a third section, the second objective of this project is underlined, with the aim to determine a method for the linear acceleration observations. Three methods are used to obtain results: the classical equations of motion, a fuzzy logic method and a linear method based on neural network optimisation. Upon analysing the results, the linear method was the best for helicopter modelling.

Research was also performed in the same team, based on flight test data for this helicopter. The subspace method was used[13] to identify the helicopter main rotor, tail rotor and engine parameters from flight test data. Ground dynamics formulations for this helicopter was validated using landing flight tests[14].The work presented in this paper on the rotorcraft model identification and validation could further be continued by building robust controllers for consistent handling qualities[10].

2.0 FLIGHT TESTS DATA PRESENTATION

The main advantages and inherent limitations of the frequency and time-domain methods were presented[11]. Then, the frequency sweep was defined1 as the typical input for frequency-domain and the multistep input for time domain methods. Three types of modelling methods existed[12]: white box, grey box, and black box methods. The first category required the user to provide the equations necessary to set up the model. This structure was very powerful when model was theoretical but it did not give good result when the environment gave random effects. At the other extreme, the black box method could be used for any data type and without prior knowledge of the system dynamics. The weakness of this method is that the reproducibility of its results is doubtful. The grey box method is a mixture of both methods. Information about the whole system might be known, while relationships between subsystems are not known. The control inputs are not the same, but in order to properly identify a system, the manoeuvre of the helicopter in response to control inputs must provide as much information as possible about the dynamic characteristics of the helicopter. Generally, for time-domain identification, standard input signals are applied which are doublet or 3211 multi-step inputs with time widths of 3s, 2s, 1s and 1s.

The 3-2-1-1 input has a much wider spectrum compared to the spectrum of the impulse or doublet inputs. The main advantage of the 3-2-1-1 input lies in its simplicity and its ability to manually realise it. Two minor aspects of 3-2-1-1 inputs are (1) their asymmetry about the trim deflection, and as a consequence, they have non zero energy at zero frequency, and (2) the first step being of larger duration, namely three units of t may lead to motions far from the initial trim conditions before the application of following steps. These undesirable effects can be minimised by modifying the input amplitudes or by time twisting the steps. The 2-3-1-1 input prevents the vehicle from going far from the trim condition, before the application of the larger duration time step.

In order to know which pilot command is relative to a flight test, the four commands are plotted which allows concluding the primarily command used by the pilot. Indeed, it is easy to distinguish a 2-3-1-1 command from another input type. Indeed, it is easy to distinguish a 2311-command from another input type.

For frequency-domain identification, frequency sweeps are usually used. The time-domain identification was performed because of the 2311-control inputs.

The flight test data are sorted by different categories: the pilot command which can be a collective, a longitudinal, a lateral or a pedal control, the flight mission which can be a level flight, an ascending flight, a descending flight or an auto-rotational flight, the helicopter loading depending on the gross weight and the longitudinal centre of gravity (CG), the altitude which varies between 3,000ft and 6,000ft and the speed which is between 30 knots and 130 knots.

A pilot manipulates the helicopter flight controls in order to correctly fly the helicopter. As previously said, four commands are used:

- The *collective* command changes angle of all main rotor blades at the same time and independently of their positions in order to control the altitude;
- The *longitudinal* cyclic command varies the main rotor blades pitch in order to control the altitude or to move forward or backward;
- The *lateral cyclic* command varies the main rotor blades pitch in order to move sideways;
- The *anti-torque pedals* command changes the pitch of the tail rotor blades, increasing or reducing the thrust produced by the tail rotor and causing the nose to yaw in the applied pedal direction.

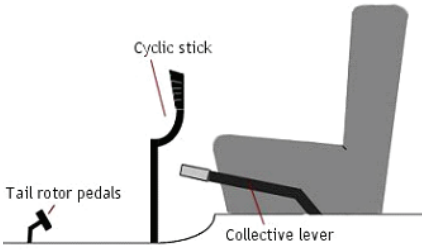

Figure 1. Localisation of the commands in a helicopter.
(http://community.bistudio.com/wiki/Image:Helicopter-controls.jpg)

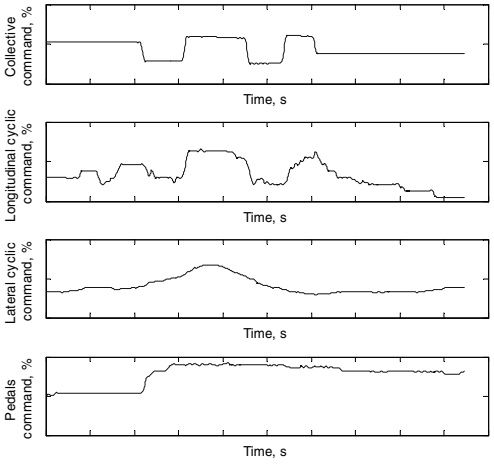

Figure 2. Pilot commands selection.

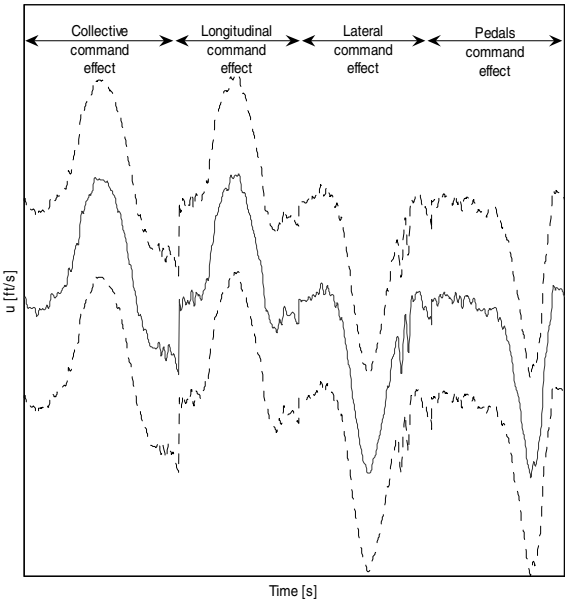

Figure 3. Example of the effect of the four concatenated commands.

Gathered data basically limits, both in terms of scope and accuracy, the model development and parameter estimation were described[6]. One of the most important aspects of data gathering is the choice of adequate inputs form to excite the aircraft motion in some optimum sense. Milliken[7] presented the optimum input as the input which excites the best the frequency range of interest. Generally, dynamic motion is excited by applying control pulse, step, multistep, or harmonic inputs. A variety of manoeuvres was usually necessary to excite dynamic motion about different axes using independent inputs on every control, such as 2311, frequency sweep, doublet inputs[6].

In Fig. 2, the flight test four controls given by the collective, the longitudinal cyclic, the lateral cyclic and pedals are plotted versus time The pilot performed a collective command, as seen in Fig. 2. The four steps of this command type are seen with different time lengths Δt for the 2s, 3s, 1s and 1s, denoted by $2\Delta t$, $3\Delta t$, Δt and Δt. The other commands are not constant during the flight test, which is due to the high correlation between all helicopter command inputs and can be explained by the fact that the aerodynamic behaviour of a helicopter cannot be split into longitudinal and lateral motion as for an aircraft study.

In order to identify the helicopter longitudinal and lateral models, a new set of data is constructed. The four pilot commands are concatenated so that for all signals, the first quarter shows the influence of the collective primary control, the second quarter shows the influence of the longitudinal cyclic control, the third quarter shows the influence of the lateral cyclic control and the last quarter shows the influence of the pedals control, as shown in Fig. 3.

In fact, the flight missions are split into four categories:

- Level flight, when the altitude rate denoted by $\dot{h}$ is between –750ft/min and +750ft/min. The altitude rate is defined by the following expression:

$$\dot{h}_{[ft/\min]} = \frac{\left[h\left(\text{flight test end}\right) - h\left(\text{flight test start}\right)\right]_{[ft]}}{\left[\text{flight test duration}\right]_{[s]}} \times 60$$

All flight tests in this category are arranged in the 'Level Flight, ±500ft/min' list.

- *Ascending flight*, when the altitude rate is higher than 750 ft/min. All flight tests in this category are arranged in the '+1,000ft/min' list.
- *Descending flight*, when the altitude rate is lower than –750 ft/min and the engines are on. All flight tests in this category are arranged in the '–1,000ft/min' list.
- *Auto-rotational flight*, when the altitude rate is lower than –750ft/min and the engines are off. All flight tests in this category are arranged in the 'Autorotation' list.

Two parameters must be known to sort the flight test data according to the helicopter loading:

- The *gross weight* which can be 'light' or 'heavy'. If the gross weight is less than 5,600lb, then the helicopter is considered to be light (*L*). If the gross weight is more than 6,000lb, then the helicopter is heavy (*H*).
- The *longitudinal centre of gravity* which can be 'aft' or 'forward'. If the longitudinal CG is more than 224 inches from the helicopter nose, then the longitudinal CG is set to the aft (*A*) of the helicopter. If the longitudinal CG is less than 220in, then it is set to the forward (*F*) of the helicopter.

Thus, four gross weight conditions are presented: (*L*), (*H*), (*A*) and (*F*). Four combinations are considered in order to study the aerodynamic behaviour of the helicopter: HA, HF, LA and LF. In this paper, the results obtained from the HA and HF flight conditions are presented for altitudes varying from 3,000ft to 6,000ft.

In order to evaluate the model's robustness, another input control is chosen. To validate the model at certain flight conditions, the control input must be related to these flight conditions.

According to the FAA, three different flight tests must be chosen to validate the estimated model. The signals must be found within the tolerance margins for at least three seconds.

The pilot flies the helicopter for different missions which are level flight, ascending, descending, and autorotation flight. Level flight tests are used to identify the models. To evaluate the quality of the models, the FAA established rules. For each variable and mission, tolerance margins are defined. In order to validate the identified model, the model is computed for another set of data in input. The model responses are compared to the measured ones. If they are 'too' different, then the model is not robust enough and the model should be again designed.

Table 1 regroups the parameters' tolerance margins for various flight missions according to the FAA rules.

Table 1
Tolerance margins for different parameters and flight missions accordingly with the FAA rules

Parameters	Variables	Flight mission	Tolerance margins	Flight mission	Tolerance margins
States	u	Level Flight Descending	± 5ft/s	Ascending	± 5ft/s
	v		± 4ft/s		± 4ft/s
	w		± 3ft/s		± 1.66ft/s
	p		± 3deg/s		± 3deg/s
	q		± 3deg/s		± 3deg/s
	r		± 3deg/s		± 3deg/s
	φ		± 1.5deg		± 1.5deg
	θ		± 1.5deg		± 3deg
Outputs	A_x		± 3ft/s^2		± 3ft/s^2
	A_y		± 3ft/s^2		± 3ft/s^2
	A_z		± 3ft/s^2		± 3ft/s^2

In order to quantify the model performance, two coefficients can be calculated: the correlation coefficient which defines the trend of a signal and the *fit coefficient* which measured the error between the measured and the estimated signals.

The correlation coefficient

The first method used to validate the model is the correlation coefficient. The correlation coefficient *Corr* (which has values between –1 and 1) is given by the following equation:

$$R = \frac{Cov\left(y,\hat{y}\right)}{\sqrt{Var\left(y\right)Var\left(\hat{y}\right)}} \quad \ldots (1)$$

Where *Cov* is the covariance, *Var* is the variance, y_s is the measured output and $\hat{y}$ is the estimated output.

The correlation coefficient R equal to one ($R = 1$) denotes perfect linear dependency (no scatter) between the measured and the calculated or estimated outputs. A correlation coefficient equal to minus one ($R = -1$) denotes *inverse linear dependency* between the measured and the estimated outputs. A correlation coefficient of zero ($R = 0$) denotes a *linear independency* between the measured and the estimated outputs. The correlation coefficient computes the goodness of the model in a statistical sense, but provides little information about model error. More information about model error can be obtained by calculation of the *fit coefficient*.

The fit coefficient FIT

The fit coefficient is used to define the fit of the estimated signals compared to the measured signals. This coefficient is defined as follows:

$$FIT = 100(1 - U) \quad \ldots (2)$$

Where:

$$U = \frac{\sqrt{\frac{1}{s-1}\sum_{i=1}^{s-1}\left(\hat{y}_i - y_i\right)^2}}{\sqrt{\frac{1}{s-1}\sum_{i=1}^{s-1}\left(\hat{y}_i\right)^2} + \sqrt{\frac{1}{s-1}\sum_{i=1}^{s-1}\left(y_i\right)^2}} \quad \ldots (3)$$

In Equation (3), the variable y_s represents the estimated signal, i.e. the model output, y is the measured or the real signal and s is the signals length. The fit is expressed as a percentage, and the U coefficient represents the ratio of the root-mean-square fit error and the root-mean-square values of the estimated and measured signals summed together. Its value is always found to be between 0 and 1 where 0 indicates a perfect fit and 1 the worst fit.

3.0 IDENTIFICATION OF THE STATE EQUATION IN OPEN-LOOP USING THE REGRESSION METHOD

The parametric fully coupled 6 DOF linear models are considered appropriate for rigid body dynamics description[2]. Generally, a state space system with discretised dynamics is defined with the following equation:

$$x(k+1) = A(k)\,x(k) + B(k)\,u(k) \quad \ldots (4)$$

Where is the state vector (**x**) and $u \in \Re^m$ is the input vector (**u**), which is the pilot commands vector. Matrices **A** and **B** are the 'state matrix' and the 'input matrix', respectively.

The state variables x are the subset of system variables that can represent the entire state of the system at any given time. The state variables are chosen to be the linear and angular velocities *u, v, w, p, q* and *r* and the Euler angles of the helicopter around the *X*-axis and *Y*-axis[2], which are denoted by φ and θ. These variables are sufficient to identify the rigid body dynamics of parametric fully-coupled 6 DOF (Degree of Freedom) models. The heading angle ψ is dropped because it does not influence a helicopter's dynamic response.

The *input* variables u is the pilot controls. The pilot can command four different flight controls: the collective pitch control which changes the angle of all of the main rotor blades at the same time and independently of their positions, the longitudinal cyclic control which varies the main rotor blades' pitch in order to control the altitude or to move forward or backward, the lateral cyclic control which varies the main rotor blades' pitch in order to move sideways, and the anti-torque pedals control which changes the pitch of the tail rotor blades, increasing or reducing the thrust produced by the tail rotor and causing the nose to yaw in the direction of the applied pedal.

Two methods are used to determine the **A** and **B** matrices. First, a *regression* method is set up to define the matrices in open-loop, and then an *optimisation* procedure is considered in order to obtain the optimal matrices in closed-loop.

For a discrete-time study, the states at a sample time depend on the states and the input controls at the previous sample time so that a regression method can be used to estimate the **A** and **B** matrices. As above mentioned, the state vector x and the controls vector u are defined as follows:

$$x = [u\ v\ w\ p\ q\ r\ \phi\ \theta]^T \quad \ldots (5)$$

$$u = [coll\ \ long\ \ lat\ \ ped]^T \quad \ldots (6)$$

A number of s equations with $(n + m)$ parameters should be estimated. The n number of inputs is n and the number of outputs is m. The k^{th} equation defining the state element x_i at step time $k + 1$ is: $\forall k \in [\![1;s]\!], \forall i \in [\![1;n]\!]$,

$$x_i(k+1) = \sum_{j=1}^{n} a_{ij} x_j(k) + \sum_{j=1}^{m} b_{ij} u_j(k) \quad \ldots (7)$$

The input vector is defined as follows:

$\forall k \in [\![1;s]\!]$,

$$(k) = [x(k)\ u(k)]^T \quad \ldots (8)$$

Where **io** is a vector with dimensions (n x 1). Details on the state equation in open loop using the regression method are given in Appendix, Equations (1) to (29).

Figure 4 shows the simulation scheme of the state equation in open loop:

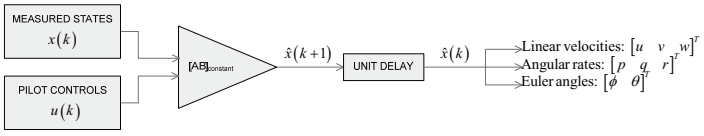

Figure 4. Simulation of the state equation in open-loop.

The plots of the states variables for one case study defined as HA6000ft-50knots are presented. The signals used to perform the model identification are set up with four 2311 concatenated pilot commands. The three validations used to validate the model are:

- Validation 1: a 2311-longitudinal command in level flight,
- Validation 2: no command in a – 1,000fpm flight, and
- Validation 3: a 2311-lateral command in level flight.

Figures 5 to 12 show the pilot command used to identify and validate the model for this flight condition (left column) and the state variables evolution (right column).

For the Figs 5, 7, 9 and 11 on the left side, the full blue line are the pilot commands used for the identification and the dashed green lines are the pilot commands used to validate the model. In Figs 6, 8, 10 and 12 on the right side, the full red lines are the estimated states signals and the dashed black lines are the tolerance margins.

Graphically, the estimated signals in the open-loop study respect the FAA tolerance margins throughout all of the time histories. The fit and correlation coefficients for this flight condition are shown in Table 2.

Table 2
Model performances for the flight condition HA6000ft-50kts

	u	*v*	*w*	*p*	*q*	*r*	φ	θ
Fit, %	99·21	99·51	97·17	99·42	99·19	99·55	99·49	99·38
Correlation,%	99·99	100	99·84	99·99	99·99	100	100	99·99

For this flight condition, the numerical results are very good as shown by the fit and correlation coefficients values that are higher than 97%. This model is identified and validated according to the FAA tolerance margins rules.

Moreover, all the estimated signals stay within the tolerance margins defined by the FAA, which increases the model goodness.

4.0 IDENTIFICATION OF THE STATE EQUATION IN CLOSED-LOOP BY USE OF AN OPTIMISATION PROCEDURE

The **A** and **B** matrices found in open-loop systems do not guarantee the good results obtained in closed loops. Indeed, a system is stable if and only if the eigen-values are negative if the system is continuous, or if and only if the eigen-values are inside a unitary circle centreed on the origin of the plan if the system is discrete. When the eigenvalues do not satisfy these constraints, then the model is not stable, the system tends to oscillate and the responses tend to diverge. An optimisation procedure is necessary to obtain good matrices in closed-loop. The open-loop system matrices are used to provide an initial guess for the optimisation.

The procedure is based on the neural networks theory. A model can be identified by providing a set of examples, i.e. input/target

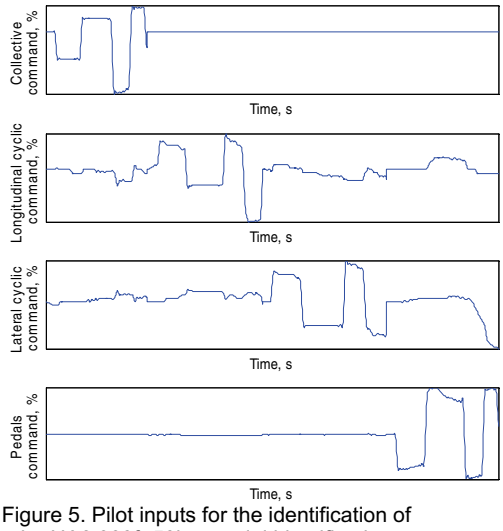

Figure 5. Pilot inputs for the identification of the HA6,000ft-50kts model identification.

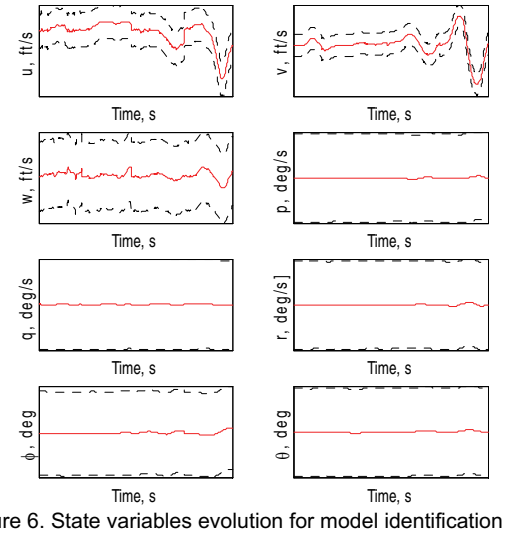

Figure 6. State variables evolution for model identification in open-loop.

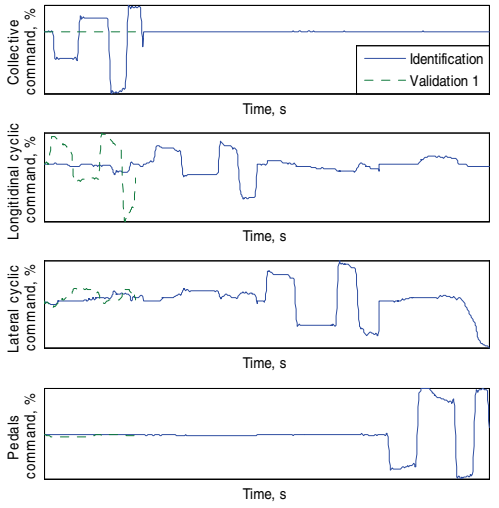

Figure 7. Pilot inputs for the identification of the HA6,000ft-50kts model identification.

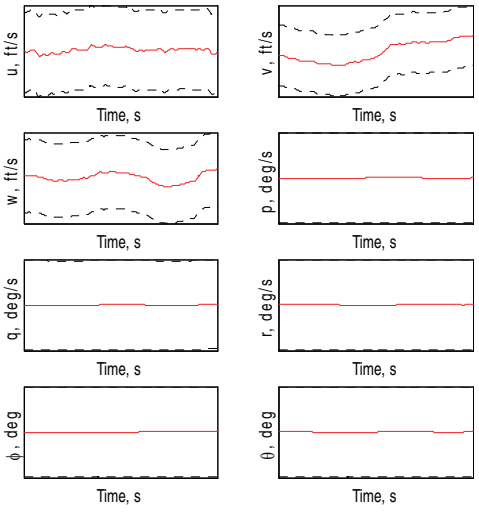

Figure 8. State variables evolution for model identification in open-loop.

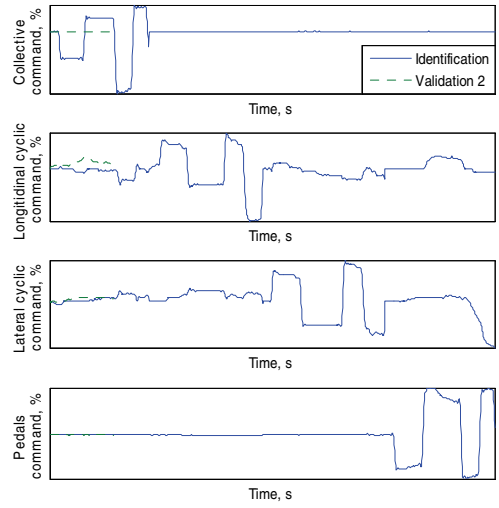

Figure 9. Pilot commands for validation 2.

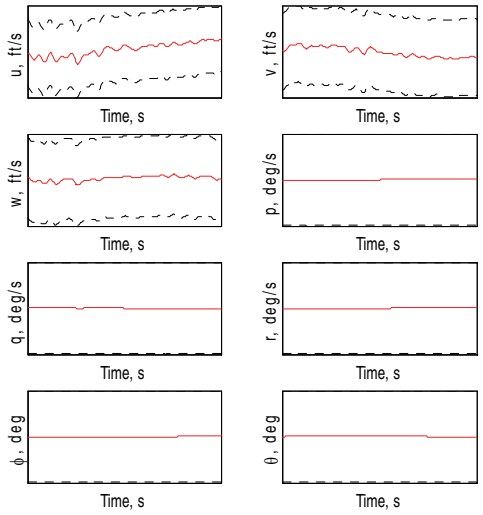

Figure 10. State evolutions in open-loop for validation 2.

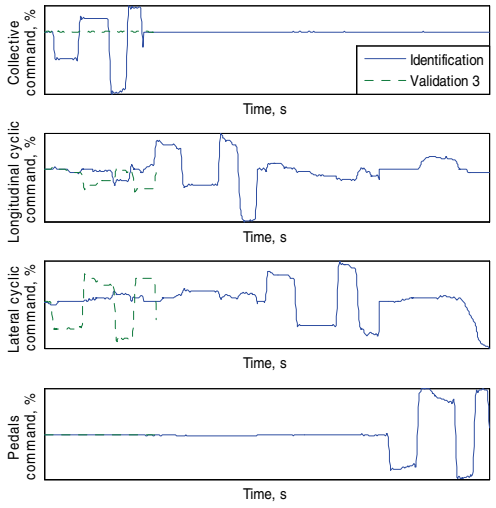

Figure 11. Pilot commands for validation 3.

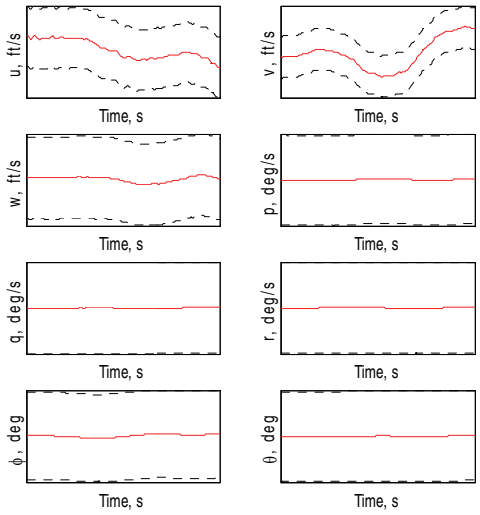

Figure 12. State evolutions in open-loop for validation 3.

pairs of proper system behaviour[3]. A neural network is composed of elements operating in parallel. The values of the connections between them are adjusted based on a comparison of the network output and the target output. This method is called backpropagation and the training is based on the Levenberg-Marquadt algorithm.

Another method to satisfy the project constraints is to tune the model by adjusting the initial conditions of the system. This technique is also called a 'proof-of-match' of the model. Thus, the neural network optimisation enables the system to be stable and the 'proof-of-match' enables the estimated state parameters to match with the measured ones.

A neural network is composed of an input p, a weight W, a bias b and a transfer function f called an activation function as shown in Figure 13:

The mathematical relation between each of these parameters can be formulated as in Equation (9):

$$a = f(Wp + b) \quad \ldots (9)$$

In order to obtain an output a with Q elements, i.e. a dimension [Q x 1], Q neurons should be arranged in parallel. Therefore, the network can be composed by several layers in series, with different activation functions. The only constraint for these functions is that they are to be differentiable. For a multilayer network, Equation (9) must be adapted in order to consider all of the activation functions. For example, the equation relating the input and output of a two-layer network (m = 2) with only one neuron (Q = 1) is:

$$\begin{Bmatrix} A_x \\ A_y \\ A_z \end{Bmatrix} = \begin{Bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{Bmatrix} + \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} \times \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \quad \ldots (10)$$

A neural network with several outputs and several layers is presented in Fig. 14:

The back propagation algorithm is a gradient descent optimisation procedure in which a mean square error performance index is minimised by adjusting the network aparameters (weights and biases). For more details, please see Equations (30) – (60) in Appendix 1.

Manual tuning is done when initial conditions are null. Empirical results are found for a majority of flight conditions: a slightly change of the Euler angles initial conditions has a great influence on the linear velocities, which can be explained by the equations of motion in which the u, v, w evolutions are influenced by θ and φ[15]. Table 3 shows the reduction of the points outside the tolerance margins when a manual tuning is done. The initial guesses are the initial conditions set to zero.

Table 3
Influence of the initial state variables on the number of points outside the tolerance margins when a manual tuning is done (%)

Flight condition	HA3,000ft	HA6,000ft	HF3,000ft	HF6,000ft
Reduction of the points outside of the tolerance margins when initial conditions are optimised,%	20·51	20·13	60·96	62·20

Table 3 shows that the initial conditions of the state equation have a strong influence on the percentage of points found outside the tolerance margins. For the HF flight conditions, the percentage of reduction is greater than 60%, which is not at all negligible.

After applying an optimisation to the open-loop models and adjustment of the initial states values, the state evolutions in closed-loop were found and represented in Figs 15 to 18.

In Figs 15 to 18, it can be seen that the estimated signals remain within the tolerance margins during all time histories. The identified model is validated with three different flight tests for this flight condition.

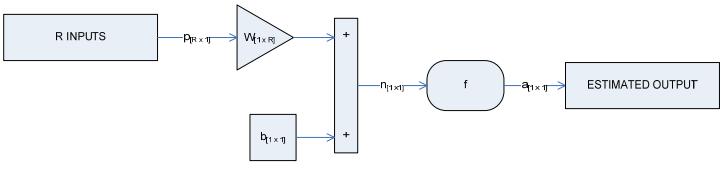

Figure 13. Neural network architecture.

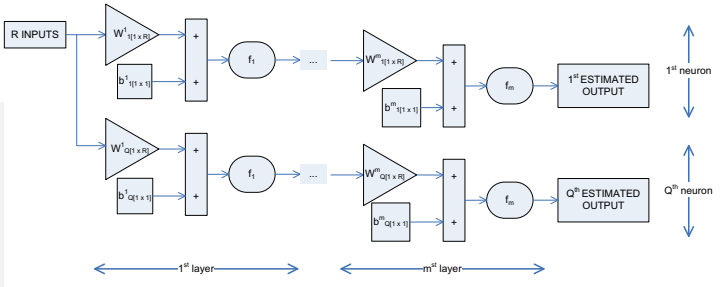

Figure 14. Multi-output and multilayer neural network architecture.

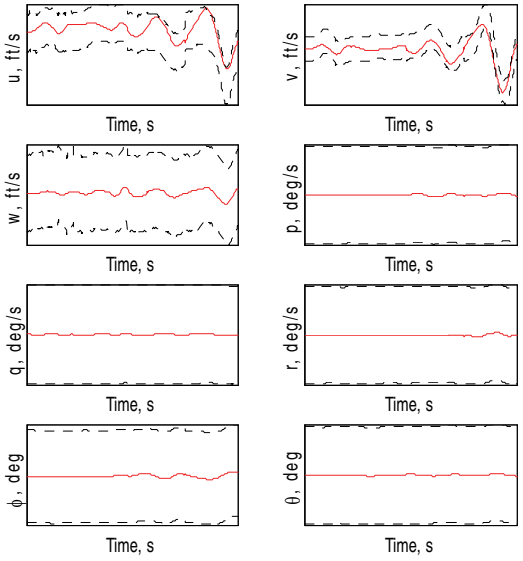

Figure 15. State evolutions in closed-loop identification.

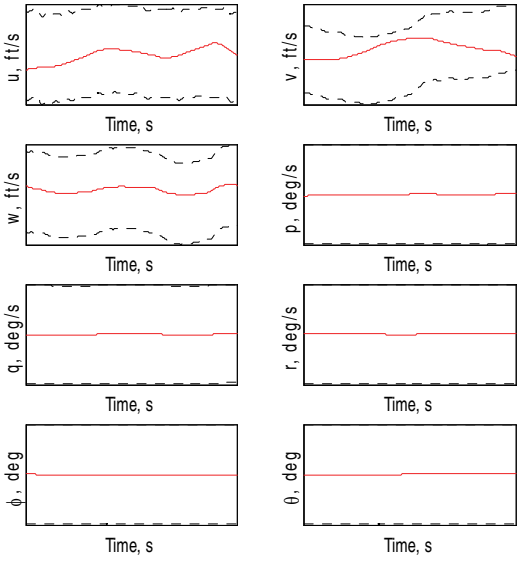

Figure 16. State evolutions in closed-loop for validation 1.

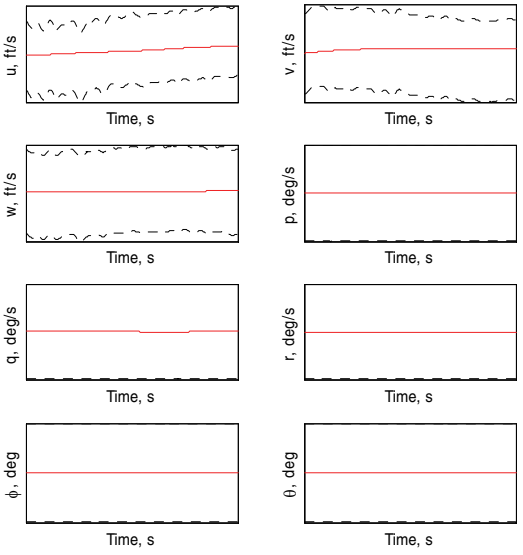

Figure 17. State evolutions in closed-loop for validation 2.

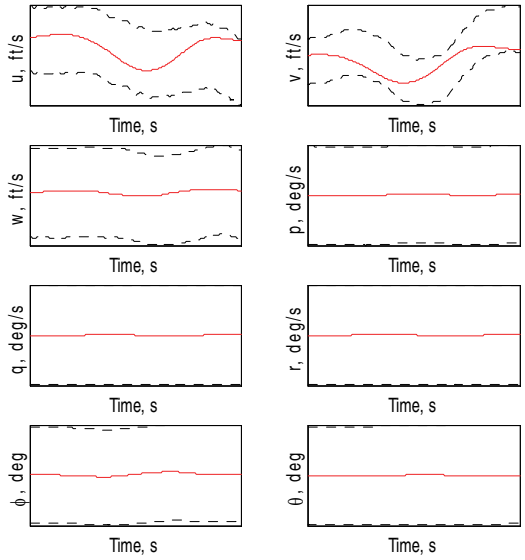

Figure 18. State evolutions in closed-loop for validation 3.

4.1 All flight conditions synthesis

In order to synthesise the obtained results for all flight conditions, an analysis should be performed on the tolerance margins to be respected during manoeuvres. In Table 4, results are shown for the identification and the three validations.

The first column presents the flight condition dependent on the helicopter loading, its altitude and speed. The second column shows the studied parameters which are the mission type, the control lengths in seconds and the tolerance margin values to be respected in open-loop (*OL*) and in closed-loop (*CL*). The other columns are the summary of the plots obtained in open- and in closed-loop.

For the flight mission type, two parameters are given: the command excited by the pilot, that can be the collective, the longitudinal cyclic, the lateral cyclic or the pedals command, and the arrows which symbolise the type of mission. *Level flight* (±500fpm) is represented by →, *ascending flight* (+1,000fpm) by ↓ and *descending flight* (-1,000fpm) by ↓. Autorotation is represented by the term 'Auto rot'.

For each case, the number of signals that are within the tolerance margins are presented within brackets [] with the instant when the signals go out of the tolerance margins. For example, [8 / 8] represents a perfect match for the model outputs versus the required FAA tolerance margins while [7 / 8] shows that one signal output (8-7) is outside the FAA tolerance margins.

By analysing results shown in Table 4, it can be observed that in open-loop, the estimated signals remain within the tolerance margins during all the controls duration. For each case, in open-loop, the model identified with 2311-command inputs is validated three times and the estimated signals are closer to those measured by more than three seconds. The FAA rules are respected in open-loop.

In closed-loop, for most of the flight conditions, all estimated signals are found within the tolerance margins. For the cases when measured signals are outside the tolerance margins, only one signal over eight does not stay within the tolerance margins. Yet, for all cases, the estimated signals remain within the tolerance margins at

Table 4
Synthesis of all flight conditions studied

Flight condition	Studied parameters		Identification	Validation1	Validation2	Validation3
HA3000ft 30kts	Flight mission		2311 →	2311 coll →	Void →	2311 long →
	Controls duration, s		25.1	6.22	5.36	5.98
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[7 / 8] 23.32s	[7 / 8] 4.16s	[8 / 8]	[7 / 8] 3s
HA3000ft 40kts	Flight mission		2311 →	2311 lat ↓	Void →	2311 coll ↑
	Controls duration, s		28.68	5.54	26.16	6.26
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HA3000ft 50kts	Flight mission		2311 →	2311 coll →	Void →	Void →
	Controls duration, s		23.74	5.6	23.3	8.74
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[6 / 8] 22.04s	[8 / 8]	[8 / 8]	[8 / 8]
HA3000ft 70kts	Flight mission		2311 →	2311 long ↑	Void →	2311 ped ↓
	Controls duration, s		26.76	5.66	12	5.72
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HA3000ft 100kts	Flight mission		2311 →	2311 lat ↑	Void →	Step long ↓
	Controls duration, s		27.54	5.82	22.12	4.78
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[7 / 8] 5.4s	[7 / 8] 3.52s
HA3000ft 110kts	Flight mission		2311 →	Step lat →	Void →	Step long ↓
	Controls duration, s		24.24	6.68	20.42	4.38
	Tolerance margins	OL				
		CL	[8 / 8]	[8 / 8]	[7 / 8] 5.4s	[7 / 8] 3.18s
HA6000ft 50kts	Flight mission		2311 →	2311 long →	Void ↓	2311 lat →
	Controls duration, s		27.16	5.58	3.92	6.84
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HA6000ft 70kts	Flight mission		2311 →	2311 long ↑	2311 ped →	2311 lat →
	Controls duration, s		25.24	6.66	5.98	8.58
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[7 / 8] 5.66s	[8 / 8]	[8 / 8]
HA6000ft 80kts	Flight mission		2311 →	2311 lat ↓	2311 coll →	2311 ped ↓
	Controls duration, s		29.7	7.1	6.6	6.96
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[7 / 8] 27.06s	[8 / 8]	[8 / 8]	[7 / 8] 3.9s

HA6000ft 90kts	Flight mission		2311 →	2311 coll →	2311 long ↑	2311 ped →
	Controls duration, s		27.92	7.9	6.42	6.08
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 30kts	Flight mission		2311 →	2311 long →	Void →	2311 lat →
	Controls duration, s		25.42	8	7.1	6.72
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 40kts	Flight mission		2311 →	2311 lat →	Void →	2311 ped →
	Controls duration, s		28.82	7.04	7.74	7.16
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 50kts	Flight mission		2311 →	2311 lat →	Void →	2311 long →
	Controls duration, s		27.4	6.02	7.36	6.14
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 60kts	Flight mission		2311 →	Step coll ↓	Doub. Long →	Step coll →
	Controls duration, s		25.56	3.82	4.32	4.32
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 70kts	Flight mission		2311 →	2311 lat →	Void →	2311 long →
	Controls duration, s		28.68	7.08	5.62	5.04
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 90kts	Flight mission		2311 →	2311 lat →	Void →	2311 long →
	Controls duration, s		27.54	6.4	14.06	6.5
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF3000ft 115kts	Flight mission		2311 →	Void →	Doub. Long →	Step lat ↓
	Controls duration, s		26.6	4.64	4.48	5.36
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF6000ft 40kts	Flight mission		2311 →	Void →	2311 ped Autorot.	2311 lat Autorot.
	Controls duration, s		30.2	3.82	8.26	8.28
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[7 / 8] 5.86s	[8 / 8]
HF6000ft 50kts	Flight mission		2311 →	2311 coll →	Void ↓	2311 long →
	Controls duration, s		32.18	7.74	6.76	10.66
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF6000ft 70kts	Flight mission		2311 →	2311 coll →	Void ↓	2311 lat →
	Controls duration, s		32.26	9.36	4.84	8

	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF6000ft 80kts	Flight mission		2311 →	2311 coll Autorot.	2311 ped ↓	2311 lat ↓
	Controls duration, s		35	7.54	7.06	7.52
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
HF6000ft 90kts	Flight mission		2311 →	2311 long →	2311 lat ↑	2311 ped →
	Controls duration, s		33.64	8.5	7.74	8.78
	Tolerance margins	OL	[8 / 8]	[8 / 8]	[8 / 8]	[8 / 8]
		CL	[8 / 8]	[8 / 8]	[8 / 8]	[7 / 8] 6.72s

least during the first three seconds. The FAA rules are also respected in closed-loop.

Thus, with the obtained results, it was found that the dynamic behaviour of the helicopter can be described by a linear state space model.

5.0 OUTPUT EQUATION DETERMINATION

Generally, in order to obtain the linear accelerations from the state parameters, the following classical equations of motion are used:

$$\begin{Bmatrix} A_x \\ A_y \\ A_z \end{Bmatrix} = \begin{Bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{Bmatrix} + \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} \times \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \quad \ldots (11)$$

Very good results were not obtained when Equation (11) was applied. In Fig. 19, the plots of the linear accelerations are traced for the flight condition: Heavy Aft flight, altitude at 6,000ft and speed of 50kts. In each plot, the solid green line corresponds to the classical method results and the dashed black lines are the tolerance margins:

Graphically, it is observed that the signals from the classical method do not remain within the tolerance margins during all time histories. Table 5 presents the average of the percentage of points outside the tolerance margins for all flight conditions when classical equations are used to observe the linear accelerations:

Table 5
Percentage of points outside the tolerance margins for all flight conditions when classical equations are used to observe the linear accelerations

	A_x	A_y	A_z
HA3,000ft	29·39	63·41	45·24
HA6,000ft	12·11	39·39	22·96
HF3,000ft	24·52	47·45	18·73
HF6,000ft	20·34	43·94	13·31

By analysing results shown in Table 5, it is noted that the classical method is not appropriate to observe the linear accelerations calculated from the state variables. The worst results are obtained when observing the linear accelerations A_y. Therefore, the theoretical model is limited, which can be explained by the fact that the signals are measured and their measures can be corrupted by instruments such as accelerometers that are not located exactly at the helicopter's centre of gravity, and by the marginal vibrations.

Thus, another method was chosen to observe the desired outputs. For this reason, two different methods were chosen: first, a *nonlinear* method, and then, a *linear* method. In both methods, the same inputs were used as the ones in the classical equations. In a discrete study, the states variables at sample times k and $k + 1$ were considered.

The fuzzy logic method is used to identify the MISO open loop models. Used to represent highly nonlinear relationships between input and output data, this algorithm can be split into two parts: numerical data clustering, and equation coefficients relating the input to the output data.

The purpose of the numerical data clustering is to define subsets so that data in the same subset are as similar as possible. Each point has a degree of belonging to clusters. Data clustering is used for statistical data analysis. This process summarises data in order to produce a concise system behaviour representation. Two parameters are used: the clusters' centres and the standard deviation of the input and output signals.

Parameter definition

In order to define the clusters, all the input and output signals are needed. The input and output data are gathered in a unique variable called X:

$$X = \left[X_{in} \quad X_{out} \right]_{[s \times (n+m)]} \quad \ldots (12)$$

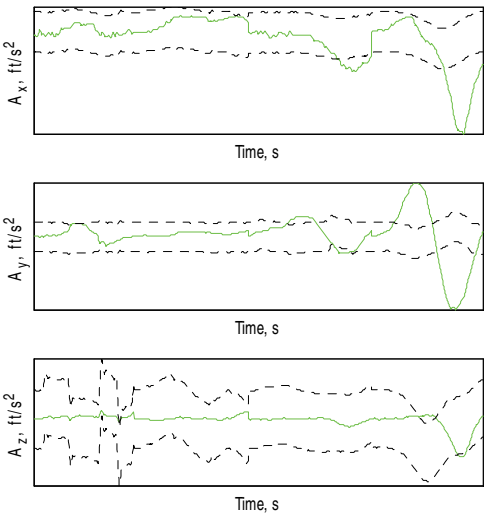

Figure 19. Output evolutions with classical equations.

Where s is the points number of the set of signals, $(n + m)$ is the number of inputs and outputs, and where X_{in} and X_{out} are the input and output signals.

Standard deviation

The standard deviation σ is defined as follows:

$$\sigma_{[s \times 1]} = \frac{radii.(maxX - minX)}{2\sqrt{2}} \quad \ldots (13)$$

Where *radii* specifies the cluster size in each data dimension, and where *minX* and *maxX* are the minima and maxima of the input and output signals. Because all signals have the same power, *radii* is fixed as the same constant for all data. Thus, a value of the standard deviation is obtained for all input and output data.

The centres of clusters

The algorithm steps are the following: signal normalisation; calculation of the potential value for each sample time; selection of the highest potential, called the reference potential; and comparison of the highest potential value with the reference value. Depending on this ratio value, is concluded the validity of this point as a cluster. Finally, during the fifth and last step, the new potential values will be calculated and the same steps as previously will be performed, until all potential values decrease to zero.

In order to define the true relationships between the input and output data, these data are normalised in the same range of values, i.e. the signals are bounded between the two values. Since no specific range is specified, the minima and maxima values are used to normalise data:

$$X_{normalized} = \frac{X_{real} - \text{minX}}{\text{maxX} - \text{minX}} \quad \ldots (14)$$

If the minima and maxima values have a zero range, a small range relative to these data is calculated, which is different from zero:

$$maxX_{new} = maxX_{old} - 0 \cdot 0001 \times \left(1 + \left|maxX_{old}\right|\right) \quad \ldots (15(a))$$

$$minX_{new} = minX_{old} - 0 \cdot 0001 \times \left(1 + \left|minX_{old}\right|\right) \quad \ldots (15(b))$$

The normalisation is validated by use of the following three 'if-then' rules:

$$if\ X_{old} < 0,\ then\ \ X_{new} = 0 \quad \ldots (16(a))$$

$$if\ X_{old} < 1,\ then\ \ X_{new} = 1 \quad \ldots (16(b))$$

$$if\ \ 0 < X_{old} < 1,\ then\ \ X_{new} = X_{old} \quad \ldots (16(c))$$

Each data point could be a cluster centre[4], therefore the potential of each data point is calculated with a measure of the likelihood that each data point has with its surrounding data points. Evaluating a cluster centre by considering that each data point could be a cluster centre is called subtractive clustering. The potential values for each sample time are calculated with following formulas:

$\forall i \in [\![1;\ s]\!]$,

$$potVals(i) = \sum_{i=1}^{s} e^{-4 \sum_{j=1}^{(n+m)} (dx_{i,j})^2} \quad \ldots (17)$$

Where:

$$dx = \begin{bmatrix} X(i,1) - X(1,1) & \ldots & X(i,(n+m)) - X(1,(n+m)) \\ \ldots & \ldots & \ldots \\ X(i,1) - X(s,1) & \ldots & X(i,(n+m)) - X(s,(n+m)) \end{bmatrix}$$

Thus, because of the negative exponential, the higher the potential value is, the closer the point selected at the i^{th} sample time is to all points of the X signals.

After calculation of all the potential values, the maximum potential value is selected; its value is defined by *maxPotVal* and its index by *maxPotIndex*. For the first cluster, the highest potential value is called *refPotVal*. Then, the maximum potential value *maxPotVal* and its index *maxPotIndex* are associated to the signals points that have the same indices:

$$maxPoint = X(\text{maxPotIndex},:) \quad \ldots (18)$$

The highest potential value is associated with the following ratio:

$$maxPotRatio = \frac{maxPotVal}{refPotVal} \quad \ldots (19)$$

For the first cluster, the highest potential value is assigned to the reference potential value so that the ratio is equal to one. The other potentials are lower than the reference potential value, so that the ratio decreases to zero and another cluster could be found.

Because the ratio decreases and tends to zero, the limits are fixed in order to accept or reject a point as a cluster. Two thresholds are introduced[4]: the *accept* ratio for the potential above which the data point will be definitively accepted as a cluster centre and the *reject* ratio for the potential below which the data point will be definitively removed from the cluster centre's list. If the ratio falls in the region between the two thresholds, a new minimum distance *minDist* is defined in order to determine if the data point has a potential high enough and far enough from the previous cluster centres:

$$minDist = \sqrt{dxSq} \quad \ldots (20)$$

Where:

$$dxSq_{[1x1]} = ((maxPoint - centers(i,:)). \times accumMultp)^2 \quad \ldots (21)$$

If $maxPotRatio + minDist < 1$, then the point is not a cluster and the potential value is set to zero. Otherwise, this point is added to the cluster's list. The new potential values are functions of the previous potential values as shown in the following equation:

$$potVals_{new} = potVals_{old} - maxPotVal_{old} \times e^{-4 \sum_{i=1}^{(n+m)} (ddx_{i,j})^2} \quad \ldots (22)$$

Where:

$$\forall l \in [\![1; \text{numPoints}]\!], \quad \ldots (23)$$

$$\begin{cases} maxP(l) = maxPoint \\ ddx = (maxP - X).new\_sqshMultp \end{cases}$$

A part of the previous maximum potential value is subtracted from each data point as a function of its distance from the previous cluster centre. With the use of the negative exponential, the point near the previous cluster centre will have a greatly reduced potential and will not be used as a cluster centre. The factor *new_sqshMultp* will represent the distance from a centre and will enable a high potential reduction.

When the potential of all data points has been updated with Equation (22), all negative potential values are arranged to zero and a data point with the highest remaining potential values is selected. The loop is realised until all potential values become equal to zero. This algorithm part is summarised in Fig. 20:

Table 6
Percentage average of the number of points outside the tolerance margins for the output equation (%)

		HA3,000ft			HA6,000ft			HF3,000ft			HF6,000ft		
		A_x	A_y	A_z	A_x	A_y	A_z	A_x	A_y	A_z	A_x	A_y	A_z
Output equation	Classic	29·39	63·41	45·24	12·11	39·39	22·96	24·52	47·45	18·73	20·34	43·94	13·31
	Fuzzy	0·09	0·23	2·48	0	0	0	0	0	0	0	0	0
	Linear NN	3·28	0·58	11·82	0	0	1·68	0·75	0·13	9,614·81	0·77	0·07	4·78

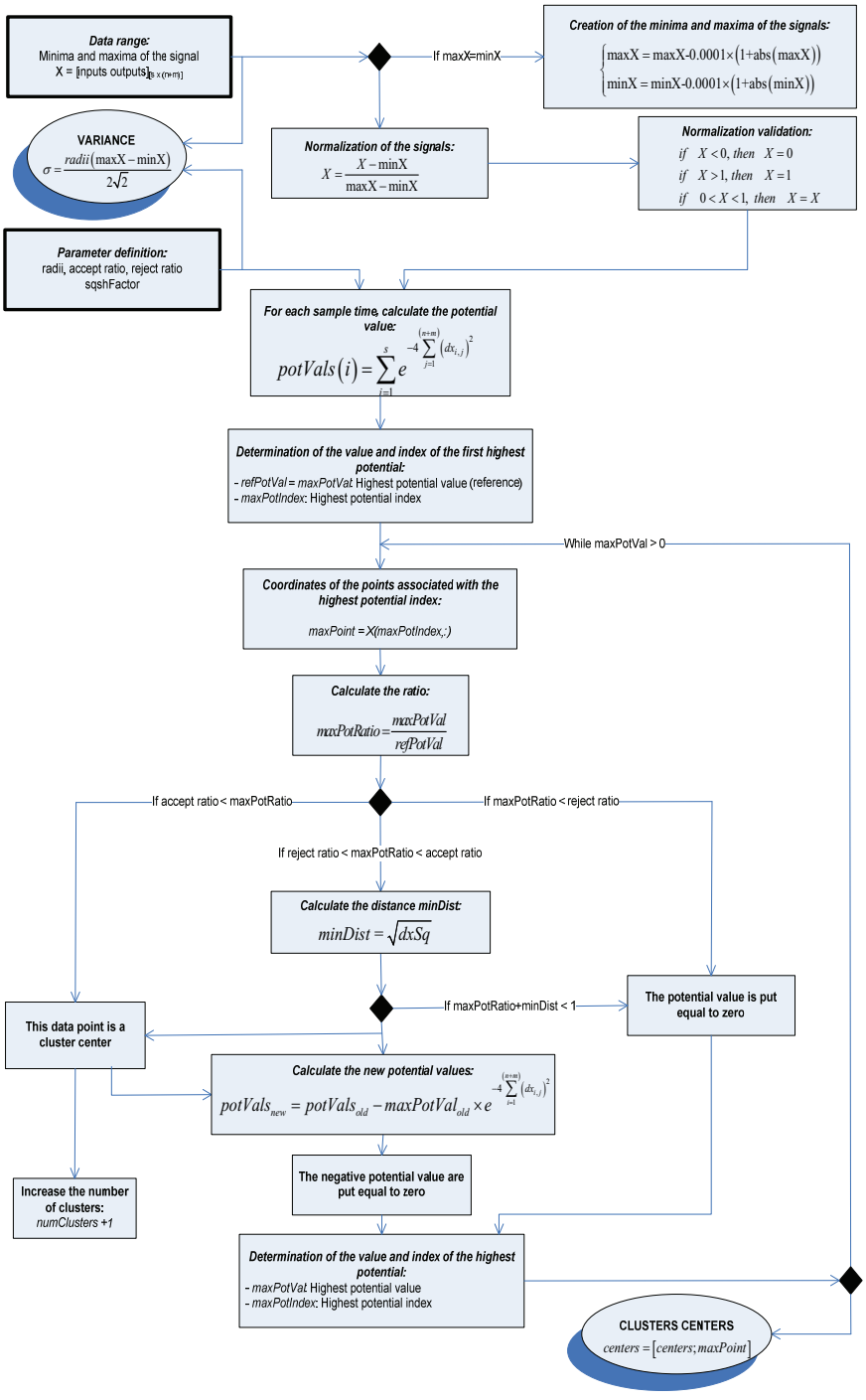

Figure 20. Subtractive clustering estimation.

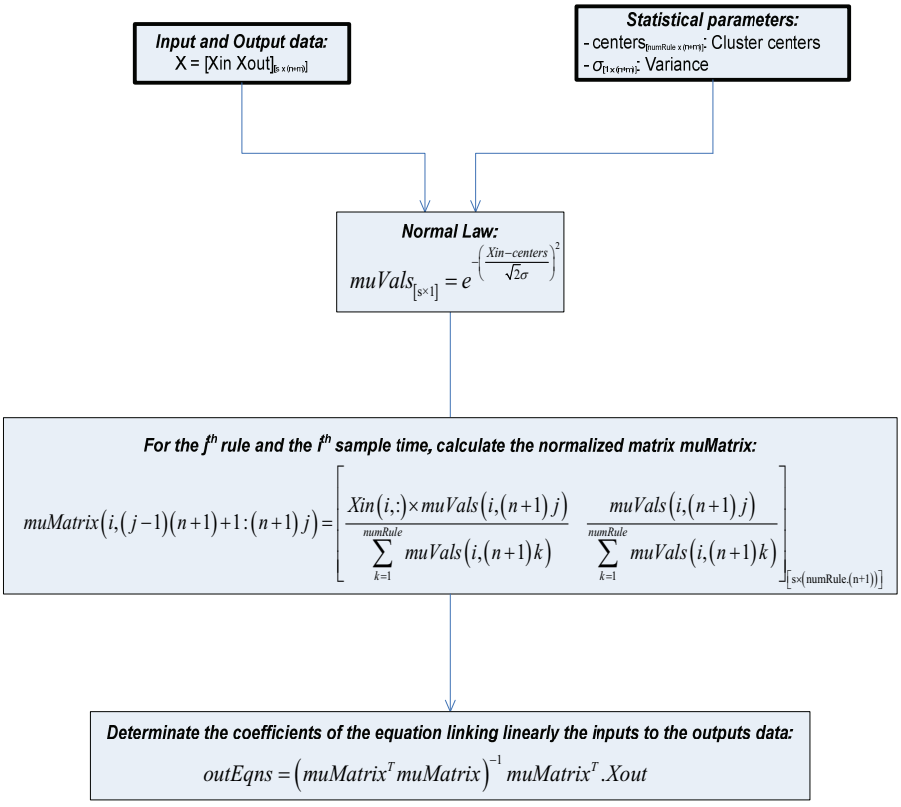

Figure 21. Estimation of the linear equation.

Each cluster centre is considered as a fuzzy rule that describes the system behaviour. In the classical form, rules or membership functions take only the value of 1 (member) or 0 (non-member). However, with fuzzy algorithms, membership functions can take infinity of values which indicate the membership of an element in a subset. Based on a negative exponential, the Gaussian membership functions give more accurate optimised models[4]. Thus, the fuzzy rules are defined as follows:

$$muVals_{[s\times 1]} = e^{-\left(\frac{Xin-centers}{\sqrt{2}\sigma}\right)^2} \quad \ldots (24)$$

Where *centres* and σ's are estimated in the first part of the algorithm. The fuzzy rules are used to define the normalised matrix *muMatrix* which represents the concordance degree between all data points and data used as centre clusters for representing the system behaviour.

$$muMatrix\left(i,(j-1)(n+1)+1:(n+1)j\right) = \left[\frac{Xin(i,:)\times muVals\left(i,(n+1)j\right)}{\sum_{k=1}^{numRule} muVals\left(i,(n+1)k\right)} \quad \frac{muVals\left(i,(n+1)j\right)}{\sum_{k=1}^{numRule} muVals\left(i,(n+1)k\right)}\right]_{[s\times(numRule.(n+1))]} \quad \ldots (25)$$

By use of this matrix, it is possible to calculate the coefficients of the linear algebraic equation relating the input to the output data. Since the *muMatrix* matrix is not square, the following equation has been used in order to obtain the coefficients:

$$outEqns = (muMatrix^T\, muMatrix)^{-1}\, muMatrix^T.Xout \quad \ldots (26)$$

These coefficients can also be obtained by the least squares method[5]. The second part of the algorithm can be summarised in Fig. 21:

Because there is a recurrence relationship between the inputs (state variables at sample time k and $k + 1$) and the outputs, the same method is used as the method above explained. No optimisation is necessary because the observed outputs depend only on the estimated states.

Independently of the state equation, the fuzzy logic and linear methods estimate the output variables using the measured states $x(k)$, as shown in Fig. 22:

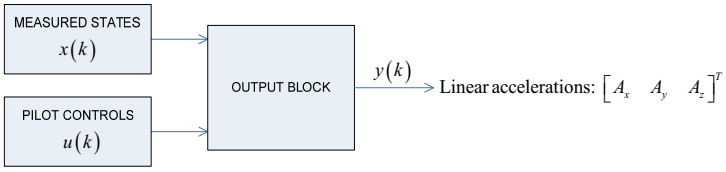

Figure 22. Simulation of the output equation.

It is observed that linear accelerations do not depend on the Euler angles, so that the Euler angles are not considered as inputs of the fuzzy and linear blocks.

Figures 23 to 26 show the estimated outputs calculated from three methods: in solid green lines, the classical equations, in solid blue lines, the linear method, and in solid red lines, the nonlinear method.

The classical equations give the worst results compared to the fuzzy logic and the linear methods. With a high percentage of points outside the tolerance margins, is denoted that using the classical equations leads to a high negative coefficient fit, which means that the estimated signals have the opposite trends of the measured signals. If pilot's impressions are taken into account, this reaction will not be appreciated. When observed outputs are estimated from the measured states signals, the fuzzy logic method gives better results than the other methods. The percentage of points outside the tolerance margins is close to zero. Thus, the identification and validation of the output block is done by use of real inputs, and the

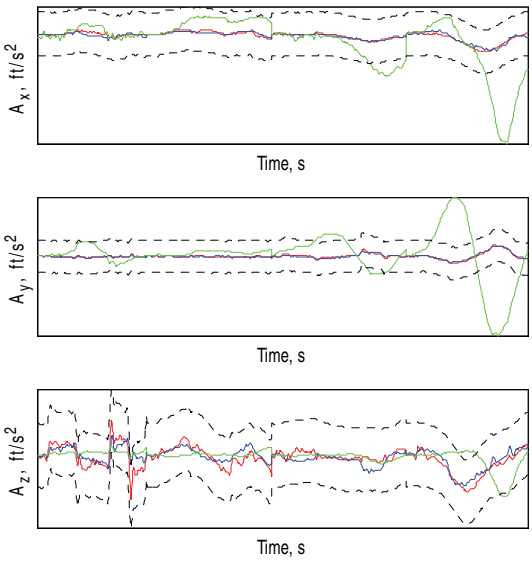

Figure 23. Outputs evolutions for identification.

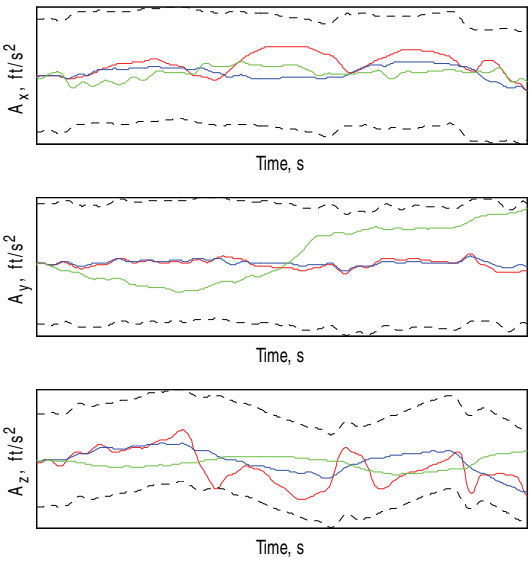

Figure 24. Outputs evolutions for validation 1.

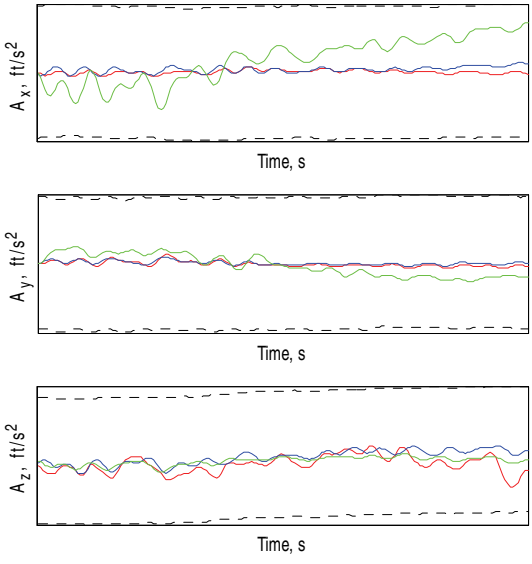

Figure 25. Outputs evolution for validation 2.

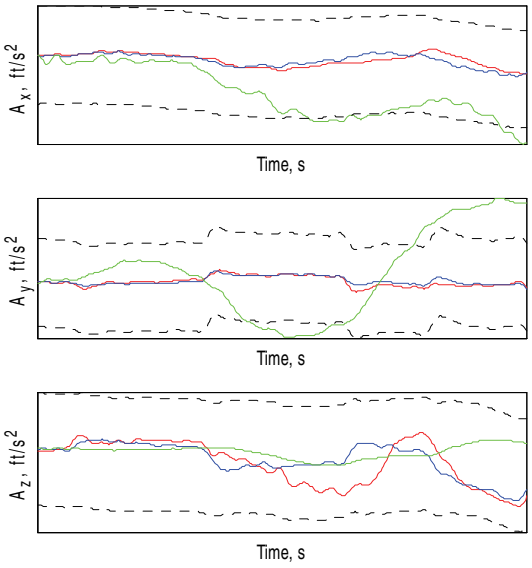

Figure 26. Outputs evolution for validation 3.

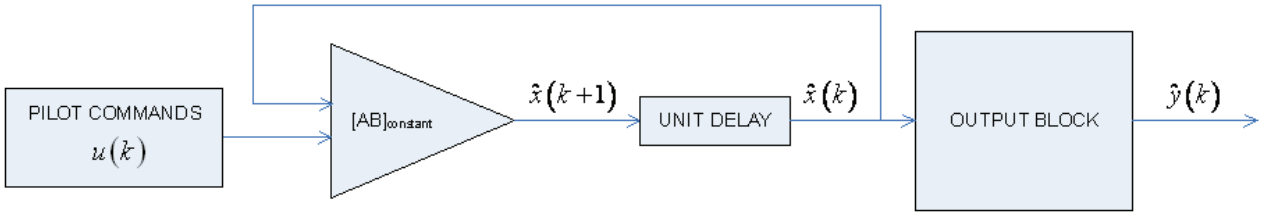

Figure 27. Simulation of the global model.

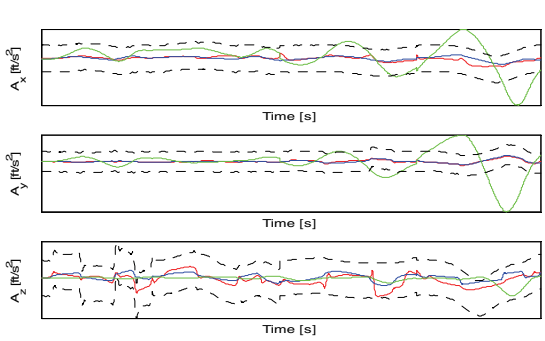

Figure 28. Output evolutions for global model identification.

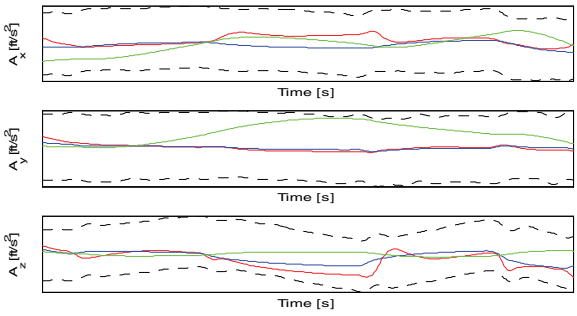

Figure 29. Output evolutions for global model validation 1.

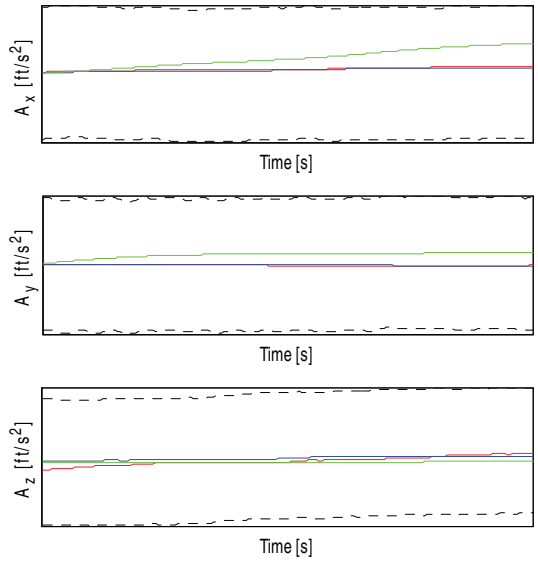

Figure 30. Output evolutions for global model validation 2.

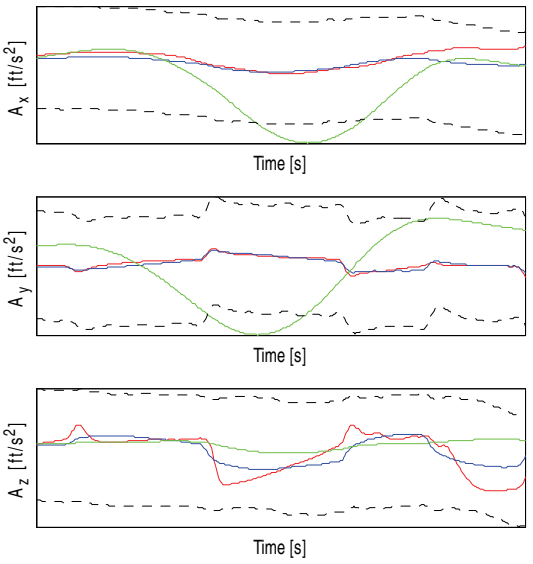

Figure 31. Output evolutions for global model validation 3.

Table 7
Percentage of the number of points outside the tolerance margins for the global model

		HA3,000ft			HA6,000ft			HF3,000ft			HF6,000ft		
		A_x	A_y	A_z	A_x	A_y	A_z	A_x	A_y	A_z	A_x	A_y	A_z
Global model	Classic	25·36	53·02	43·30	11·26	38·88	18·42	18·7	40·75	17·55	17·04	45·1	12·22
Fuzzy		7·01	2·79	31·04	0·92	0·37	12·51	0·79	0·08	4·81	1·75	0	8·6
Linear NN		3·54	0·78	22	0·04	0	3·59	0·61	0·13	4·17	0·01	0·77	6·1

fuzzy logic method seems to be the best method.

6.0 GLOBAL MODEL

The two blocks presented in sections 2 and 3 are regrouped in order to create a global model in which the observed outputs are computed from the estimated states. The following scheme is then simulated:

By use of this simulation, the robustness of the best block found in section 3 is tested to observe the linear accelerations.

Figures 28 to 31 show the estimated outputs calculated from three methods: in solid green lines, the classical equations, in solid blue lines, the linear method, and in solid red lines, the nonlinear method. The estimated outputs are the acceleration results when the output block inputs are the estimated states.

Table 7 summarises the percentage of points outside the tolerance margins for all of the studied flight conditions.

As in the output equation study, the classical equations give the worst results compared to the fuzzy logic and the linear methods in the global model. When the behaviour of the fuzzy logic block is studied, where the inputs are the estimated states signals, is noticed that the estimated outputs are not very good. Sudden jumps are observed in the estimated outputs which can be felt by a pilot in a simulator.

Contrary to the fuzzy logic method, the linear method is more stable when changing the measured states by the estimated states. This is a good argument, because each flight test is different even if it is realised with the same flight conditions. The linear block absorbs the small differences in signals. Thus, in order to properly observe the linear accelerations of the system, the linear block is more appropriate because of its robustness.

7.0 CONCLUSIONS

A helicopter model was identified for 22 flight conditions. Each flight condition was defined by an altitude (from 3,000ft to 6,000ft), a speed (from 30 knots to 115 knots), a helicopter loading (heavy or

light) and a position of the helicopter CG (aft or forward). To identify the models, 2-3-1-1 multistep control inputs were performed by the pilot to excite all helicopter modes. Four types of pilot controls were used: the collective, the longitudinal cyclic, the lateral cyclic and the pedals controls. In order to identify the longitudinal and lateral motions of the helicopter, a new data set was constructed by concatenating the data related to each of the four control input.

The state evolution from the pilot controls was determined. The state variables were the linear and angular speeds and the *x*- and *y*- axis Euler angles. This problem was treated in open-loop (the estimated state signals were defined by the pilot controls and the measured state signals) and in closed-loop (the estimated state signals were defined only by the pilot controls).

A recursive method was used to define the relationship between the states and the pilot controls. An optimisation based on neural network theory was then applied in order to increase the goodness of the model in closed-loop. Then, a tuning of the initial state conditions was implemented in order to satisfy the FAA rules. Indeed, to guarantee the goodness of a model, the FAA has defined several rules to be satisfied by the model. The first rule regarded the tolerance margins the estimated signals have to remain within. These tolerance margins were defined for each variable and for each helicopter mission (level flight, ascending flight, descending flight and autorotation). The second rule was about the model validation. All identified models must be validated by at least three different flight tests not used for the models identification and the estimated signals must remain in the tolerance margins at least during three seconds. For the 22 flight conditions, the state evolution in open-loop and in closed-loop satisfies the FAA rules so that the method to generate the state equation was validated. The first conclusion was that the dynamical behaviour of the helicopter would be defined by a six DOF linear model.

The second problem was about the outputs observation from the state variables. The outputs variables were the linear accelerations. It has been shown that the classical equations of motion did not give good results, which is explained by the fact that the theoretical equations did not cope with the random effects of the environment when data are measured. Thus, two methods were defined and a comparison between these two methods was done in order to find a powerful method to observe the system outputs from the system states: a fuzzy logic method, a linear method optimised with a neural network algorithm (linear NN). In order to do proper comparisons among results obtained with the three methods, all blocks have the same inputs and the same outputs. Then, because the model could be implemented in a simulator for the pilot training, the pilot feedback is very useful in order to compare the reality with the results of the mathematical model. Thus, the comparison among the three methods was based on the plots particularities. A list of criteria was defined with weighted coefficients. When the outputs are obtained from the measured state variables, the linear NN and fuzzy logic methods give noticeably the same results but the linear method obtained a better score.

A clear conclusion about this second problem can be given when the global model is set up. Indeed, the global model is the most realistic and could be implemented in a simulator to reproduce the reality. The outputs must be defined by the estimated state variables and not by the measured ones. The estimated states variables depend only on the pilot controls (closed loop study). With this model, the results are clearer. The fuzzy logic method is not robust

enough so that the outputs plots shows peaks, which can be felt by the pilot. A better final score is obtained by the linear method more than the fuzzy logic method.

The last step is to use the conclusions of both problems in order to generate the dynamical behaviour of the helicopter for any flight condition.

APPENDIX

The parameter vector gathers the elements of the **A** and **B** matrices and is formed by the parameters to be estimated. The elements of the ith line of **A** and **B** matrices at the k^{th} step time are denoted as follows:

$$\forall k \in [\![1;s]\!], \forall i \in [\![1;n]\!],$$

$$ab_i(k) = [a_{i1}\ a_{i2}\ a_{i3}\ a_{i4}\ a_{i5}\ a_{i6}\ a_{i7}\ a_{i8}\ b_{i1}\ b_{i2}\ b_{i3}\ b_{i4}] \quad \ldots \text{(A1)}$$

The i^{th} state at the k^{th} sample time can be written, based on Equation (7) from the paper:

$$\forall k \in [\![1;s]\!], \forall i \in [\![1;n]\!],$$

$$x_1(k+1) = ab_i(k)\ io\ (k) + e_i\ (k) \quad \ldots \text{(A2)}$$

Where *e* is the error to be minimised.

For *s* measurements:

$$\begin{aligned} x_i\ (2) &= ab_i\ (1)\ io\ (1) + e_i(1) \\ x_i\ (s) &= ab_i\ (s-1)\ io\ (s-1) + e_i(s-1) \end{aligned} \quad \ldots \text{(A3)}$$

These *s* equations are summarised with the following formulation:

$$X_{des} = AB.IO + e \quad \ldots \text{(A4)}$$

Where:

$$\begin{cases} X_{des} = [X_{des}(2)\ \ldots\ x_{des}\ (s)]^T \\ IO = [io(1)\ \ldots\ io(s-1)]^T \\ e = [e(1)\ \ldots\ e(s-1)]^T \end{cases} \quad \ldots \text{(A5)}$$

In order to minimise the error, the following cost function *J* is defined:

$$J(AB) = e^T We = \sum_{k=1}^{s} w(k) e^2(k) \quad \ldots \text{(A6)}$$

Where *W* is a weighted matrix which is diagonal. If the diagonal terms $w(k)$ are equal to one, then it means that the error coefficients have the same weight. If not, then it means that the error coefficients are different. This weighted matrix was chosen[8], so that the element on the i^{th} line and the j^{th} column is defined as follows:

$$W_{ij}\ (k) = \gamma^{s-k} \quad \ldots \text{(A7)}$$

Where $\gamma < 1$ is the forgetting factor and *s* is the measurements number.

As *s*-*k* gets larger and γ does not equal to one, the weighting factor approaches zero, therefore older points receive little weight. As *s*-*k* goes to zero, the weighting factor approaches one and the most recent data are favoured. The smaller γ is, the faster the algorithm can track, but the more the estimates vary, even the true parameters are time-invariant. The cost function is further obtained:

$$\begin{aligned} J(AB) = e^T We &= (X_{des} - AB.IO)^T W (X_{des} - AB.IO) \\ &= X_{des}^T W X_{des} - 2X_{des}^T WAB.IO + IO^T.AB^T W\ AB.IO \end{aligned} \quad \ldots \text{(A8)}$$

In order to minimise the cost function, its derivative is defined as equal to zero:

$$\frac{\partial J(AB)}{\partial AB} = -2X_{des}^t W\ IO + 2AB^t IOWAB = 0 \quad \ldots \text{(A9)}$$

and next equation is obtained:

$$X_{des}^{T} WIO = IO^{T}.AB^{T} WIO \quad \ldots (A10)$$

Transposition of Equation (A10) gives:

$$IO^{T} WX_{des} = IO^{T} W\, IO.AB \quad \ldots (A11)$$

The parameter vector is estimated as follows:

$$\hat{AB} = (IO^{T} WIO)^{-1}\, IO^{T} W.X_{des} \quad \ldots (A12)$$

The matrix $IO(N+1)$ can be written as:

$$IO(N+1) = [io(1)\ \ io(n+1)\ \ \ldots\ \ io(N+1)]^{T} \quad \ldots (A13)$$

The term $IO^{T}WIO$ of the expression of $\hat{AB}$ (see Equation (A12)) is developed by introducing a recurrence relationship:

$$\begin{aligned} IO(N+1)^{T} W(N+1) IO(N+1) &= \sum_{k=1}^{N+1} io(k) w(k) io^{T}(k) = \sum_{k=1}^{N+1} io(k) \gamma^{N+1-k} io^{T}(k) \\ &= \sum_{k=n}^{N} io(k) \gamma\gamma^{N-k} io^{T}(k) + io(N+1) io^{T}(N+1) \\ &= \gamma IO^{T}(N) W(N) IO(N) + io(N+1) io^{T}(N+1) \end{aligned} \quad \ldots (A14)$$

The matrix **P** is defined as follows:

$$P^{-1}(k) = IO^{T}(k) W(k) IO(k) \quad \ldots (A15)$$

By substituting the expression of P in Equation (A14) for $k = N$, the expression of P becomes:

$$P(N+1) = [\gamma P^{-1}(N) = io(N+1)\, io^{T}(N+1)]^{-1} \quad \ldots (A16)$$

In order to obtain a recurrence expression of P, the following analytical formula is used:

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1} DA^{-1} \quad \ldots (A17)$$

By denoting $A = \gamma P^{-1}(N)$, $B = io(N+1)$, $C = 1$, $D\ io^{T}(N+1)$

The expression of $P(N+1)$ is rewritten as follows:

$$P(N+1) = \frac{P(N)}{\gamma} - \frac{P(N)}{\gamma} io(N+1) \left[1 + io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) \right]^{-1} io^{T}(N+1) \frac{P(N)}{\gamma} \quad \ldots (A18)$$

With the same reasoning, the term $IO^{T}WX_{des}$ is given by Equation (A12):

$$IO^{T}(N+1) W(N+1) X_{des}(N+1) = \gamma IO^{T}(N) W(N).\ X_{des}(N) + io(N+1) x_{des}(N+1) \quad \ldots (A19)$$

A new expression of $\hat{AB}$ is reformulated by use of Equations (A18) and (A19):

$$\hat{AB}(N+1) = \left[\frac{P(N)}{\gamma} - \frac{P(N)}{\gamma} io(N+1) \left[1 + io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) \right]^{-1} io^{T}(N+1) \frac{P(N)}{\gamma} \right] .\left[\gamma IO^{T}(N) W(N) X_{des}(N) + io(N+1) x_{des}(N+1) \right] \quad \ldots (A20)$$

Hence, by substituting Equation (A18) into Equation (A12), the estimated parameter matrix is defined as:

$$\hat{AB}(N) = P(N)\, IO^{T})N) W(N)\, X_{des}(N) \quad \ldots (A21)$$

$\hat{AB}(N)$ given by given by Equation (A21) is replaced into Equation (A20) to obtain:

$$\begin{aligned} \hat{AB}(N+1) = {} & \hat{AB}(N) + \frac{P(N)}{\gamma} io(N+1) x_{des}(N+1) \\ & - \frac{P(N)}{\gamma} io(N+1) \left[1 + io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) \right]^{-1} io^{T}(N+1) \hat{AB}(N) \\ & - \frac{P(N)}{\gamma} io(N+1) \left[1 + io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) \right]^{-1} io^{T}(N+1) \frac{P(N)}{\gamma} io \\ & (N+1) x_{des}(N+1) \end{aligned} \quad \ldots (A22)$$

The **K** matrix is written as:

$$K(N+1) = \frac{P(N)}{\gamma} io(N+1) \left[1 + io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) \right]^{-1} \quad \ldots (A23)$$

Equation (A22) becomes:

$$\begin{aligned} \hat{AB}(N+1) = {} & \hat{AB}(N) + \frac{P(N)}{\gamma} io(N+1) x_{des} - K(N+1) io^{T}(N+1) \hat{AB}(N) \\ & - K(N+1) io^{T}(N+1) \frac{P(N)}{\gamma} io(N+1) x_{des} \end{aligned} \quad \ldots (A24)$$

The concise form of is obtained:

$$\hat{AB}(N+1) = \hat{AB}(N) + K(N+1) \left[x_{des}(N+1) - io^{T} \hat{AB}(N) \right] \quad \ldots (A25)$$

By substituting Equation (A23) into Equation (A18), the new expression of P is written as:

$$P(N+1) = \frac{1}{\gamma} \left[I_{d} - K(N+1) io^{T}(N+1) \right] P(N) \quad \ldots (A26)$$

To implement this method in Matlab, the algorithm is as follows:

- Initialisation of the forgetting factor γ so that $0 < \gamma \leq 1$, which corresponds to an exponential weighting.
- Initialisation of the matrices **P** and $\hat{AB}$ where **P** is chosen as a diagonal matrix so that its diagonal terms have high values.
- Calculation of the **K** matrix: $K(k+1) = (P(k) io(k+1)[\gamma + io(k=1) P(k) io(k+1)]^{-1}$
- Calculation of the $\hat{AB}$ matrix: $\hat{AB}(k+1) = \hat{AB}(k) + K(k+1)\, [x_{des}(k+1) - io^{T}(k+1)\, AB(k)]$
- Calculation of the **P** matrix: $P(k+1) = \frac{1}{\gamma} \left[I_{d} - K(k+1) io^{T}(k+1) \right] P(k)$
- Go to the third step and perform the iteration procedure again.

For the open-loop system, no state feedback is used. The state variables are functions of the real state variables and the pilot controls at the previous step time. The regression method previously described is used to estimate the state variables. For each time step, the parameters matrices **A** and **B** are obtained. Each pair of matrices was tested for the entire signals, and the pairs giving the best results were selected. An index was defined to characterise the matrix performance. This index is used in the fuzzy logic method to find the potential value of each point:

$$\forall \mathbf{k} \in \Re^{s}, \forall \mathbf{i} \in \Re^{n}, \quad pot(k) = \sum_{j=1}^{s} e^{-4 \sum_{i=1}^{n} \left[X_{des}(i) - A\hat{B}_{i}(k) IO(k) \right]} \quad \ldots (A27)$$

The highest this index is, the better the performance of the selected matrix. Since the matrix value was determined, the state space dynamics is time-invariant:

$$x(k+1) = Ax(k) + Bu(k) \quad \ldots \text{(A28)}$$

that can be written under the following matrix form:

$$x(k+1) = \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \quad \ldots \text{(A29)}$$

Details of the back propagation algorithm are given in Equations (A30) to (A60).

The performance index at iteration k is defined as follows:

$$F(w,b) = \frac{1}{2} e^T(k) e(k) = \frac{1}{2} \left[y_{des}(k) - a(k) \right]^T \left[y_{des}(k) - a(k) \right] \quad \ldots \text{(A30)}$$

Where y_{des} is the target output and a is the neural network output which is compared to the desired output. The steepest descent algorithm considered for the approximate mean square error modelling is:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \eta \frac{\partial \hat{F}}{\partial w_{i,j}^m} \quad \ldots \text{(A31(a))}$$

$$b_i^m(k+1) = b_i^m(k) - \eta \frac{\partial \hat{F}}{\partial b_i^m} \quad \ldots \text{(A31(b))}$$

Where m is the number of the considered layer and η is the learning rate called the network training speed. For a multilayer network, the error is an indirect function of the weights in the hidden layers, so that these derivatives are calculated differently:

$$\frac{\partial \hat{F}}{\partial w_{i,j}^m} = \frac{\partial \hat{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial w_{i,j}^m} \quad \ldots \text{(A32(a))}$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = \frac{\partial \hat{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial b_i^m} \quad \ldots \text{(A32(b))}$$

It can be written:

$$n_i^m = \sum_{j=1}^{S^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m \quad \ldots \text{(A33)}$$

So that the two derivatives $\frac{\partial n_i^m}{\partial w_{i,j}^m}$ and $\frac{\partial n_i^m}{\partial b_i^m}$ are directly defined since the net input to layer m is an explicit function of the weights and biases in that layer:

$$\frac{\partial n_i^m}{\partial w_{i,j}^m} = a_j^{m-1} \quad \ldots \text{(A34(a))}$$

$$\frac{\partial n_i^m}{\partial b_i^m} = 1 \quad \ldots \text{(A34(b))}$$

The F sensitivity is defined as any effect of the system function or any other system characteristic caused by a change in one or more system parameters. The larger the system sensitivity, the stronger is the effect of small changes on the system performances. It is defined:

$$s_i^m = \frac{\partial F}{\partial n_i^m} \quad \ldots \text{(A35)}$$

Then:

$$\frac{\partial F}{\partial w_{i,j}^m} = s_i^m a_j^{m-1} \quad \ldots \text{(A36(a))}$$

$$\frac{\partial F}{\partial b_i^m} = s_i^m \quad \ldots \text{(A36(b))}$$

Thus, the expressions of weights and biases are rewritten as follows:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \eta s_i^m a_j^{m-1} \quad \ldots \text{(A37(a))}$$

$$b_i^m(k+1) = b_i^m(k) - \eta s_i^m \quad \ldots \text{(A37(b))}$$

The Jacobian matrix is next defined:

$$\frac{\partial n_i^{m+1}}{\partial n_m} = \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix} \quad \ldots \text{(A38)}$$

One of the elements of the above matrix can written:

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial \left(\sum_{l=1}^{S^m} w_{il}^{m+1} a_l^m + b_i^{m+1} \right)}{\partial n_j^m}$$

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = w_{il}^{m+1} \frac{\partial a_l^m}{\partial n_j^m} \quad \ldots \text{(A39)}$$

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = w_{il}^{m+1} \frac{\partial f^m(n_j^m)}{\partial n_j^m}$$

By generalisation of this result:

$$\frac{\partial n^{m+1}}{\partial n^m} = W^{m+1} \dot{F}^m(n^m) \quad \ldots \text{(A40)}$$

Where:

$$\dot{F}^m = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \ldots & 0 \\ 0 & \ldots & 0 & \ldots \\ \ldots & 0 & \ldots & 0 \\ 0 & \ldots & 0 & \dot{f}^m(n_{S^m}^m) \end{bmatrix} \quad \ldots \text{(A41)}$$

The recurrence relationship between sensitivities at different layers is written by using the chain rule:

$$s^m = \frac{\partial F}{\partial n^m} = \frac{\partial F}{\partial n^{m+1}} \frac{\partial n^{m+1}}{\partial n^m} = \left(\frac{\partial n^{m+1}}{\partial n^m} \right)^T \frac{\partial F}{\partial n^{m+1}}$$

$$s^m = \dot{F}^m(n^m) \left(W^{m+1} \right)^T s^{m+1} \quad \ldots \text{(A42)}$$

Where $m = M-1, \ldots, 2$, and 1 and M are the number of layers.
The starting point s^M (where M is the output layer) is defined with the recurrence relationship between sensitivities:

$$s_i^M = \frac{\partial F}{\partial n_i^M} = \frac{\partial (t-a)^T (t-a)}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{S^M} (t_j - a_j)^2}{\partial n_i^M} \quad \ldots \text{(A43)}$$

$$s_i^M = -2(t_i - a_i) \frac{\partial a_i}{\partial n_i^M}$$

Since:

$$\frac{\partial a_i}{\partial n_i^M} = \frac{\partial a_i^M}{\partial n_i^M} = \frac{\partial f^M(n_j^M)}{\partial n_i^M} = \dot{f}^M(n_j^M) \quad \ldots \text{(A44)}$$

Next equation can be written as:

$$s_i^M = -2(t_i - a_i) \dot{f}^M(n_j^M) \quad \ldots \text{(A45)}$$

By summarising, the input is propagated forward through the network during the first step. Then, during the second step, the sensitivities are propagated back through the network. Finally, during the third step, the biases and weights are updated. When the algorithm begins to diverge, the system is oscillating back and forth. If the trajectory could be filtered by averaging the updates to the parameters, then the oscillations might be smoothed out and a stable trajectory would be produced by use of a low-pass filter. By introducing the momentum coefficient γ which corresponds to the low-pass filter effect, following equations are obtained:

$$\Delta w_{i,j}^{m}(k+1) = \gamma \Delta w_{i,j}^{m}(k) - (1-\gamma)\eta s_i^m a_j^{m-1} \quad \ldots \text{(A46(a))}$$

$$\Delta b_i^{m}(k+1) = \gamma \Delta b_i^{m}(k) - (1-\gamma)\eta s_i^m \quad \ldots \text{(A46(b))}$$

The algorithm used in neural network theory to identify system behaviour is applied here. Indeed, the system has eight states variables and four control inputs, therefore a total of 12 inputs (R = 12), please see Fig. 14. Due to the fact that matrix A has dimensions [8 × 8], the network must have eight neurons arranged in parallel (Q = 8 in Fig. 14).

The state equation is defined as Equation (A29). Equation (A29) in Appendix 1, and equation (A10) in the paper, show that (1) the biases must be null, (2) the activation function must be linear in order to ensure the relation $f(a) = a$ and (3) the network must have only one layer. The adjustment of the weights values i.e. the **A** and **B** matrices elements can be done with an optimisation on-line algorithm in which the weight values are updated. By focusing on this particular case of the neural network theory, the **A** and **B** matrices must be updated as follows:

$$A_{new} = A_{old} - dA_{new} - \mu dA_{old} \quad \ldots \text{(A55(a))}$$

$$B_{new} = B_{old} - dB_{new} - \mu dB_{old} \quad \ldots \text{(A55(b))}$$

Where:

$$dA_{new} = -2n(x_{des}(k+1) - \hat{x}(k+1))\hat{x}(k) \quad \ldots \text{(A56(a))}$$

$$dB_{new} = -2n(x_{des}(k+1) - \hat{x}(k+1))\hat{u}(k) \quad \ldots \text{(A56(b))}$$

In order to increase the on-line optimisation power, the pilot controls and desired states are concatenated. After several tests, it was observed that five cycles are sufficient to obtain good results. Then, at each sequence, the algorithm parameters (learning rate and momentum) are divided by two at each cycle in order to ensure the convergence of the optimisation algorithm.

As mentioned above, a recurrence relationship exists between the state variables of a system. For k = 0:

$$x(1) = Ax(0) + Bu(0) \quad \ldots \text{(A57)}$$

Then, for k = 1:

$$x(2) = Ax(1) = Bu(1)$$

$$x(2) = A[Ax(0) + \mathrm{Bu}(0)] + Bu(1) = \mathrm{A}^2\mathrm{x}(0) + \mathrm{A}Bu(0) + \mathrm{Bu}(1) \quad \ldots \text{(A58)}$$

For k = 2:

$$x(3) = Ax(2) = Bu(1)$$

$$x(3) = A[Ax(1) + B(1)] + Bu(2)$$

$$\mathrm{x}(3)\, A[A[A[Ax(0) + B\mathrm{u}(0) + B\mathrm{u}(1)] + Bu(2) = A^3x(0) + A^2Bu(0) + ABu(1) + Bu(2) \quad \ldots \text{(A59)}$$

and thus, for $k = n$:

$$x(n+1) = A^{n+1}x(0) + A^nBu(0) + \sum_{j=1}^{n} A^{n-j}Bu(j) \quad \text{(A60)}$$

By analysing this equation, two ways to influence the state evolution are possible: tuning of the initial state values $x(0)$ and slight modification of the initial command inputs $u(0)$. For system identification, it is more acceptable to tune the initial state variables than the command inputs. For validation, both tuning parameters can be used.

REFERENCES

1. TISCHLER, M.B and REMPLE, R.K. *Aircraft and Rotorcraft System Identification: Engineering Methods with Flight Tests Examples*, AIAA Education Series, AIAA, ISBN-10: 1-56347-837-4, 2006, pp 1-600.
2. HAMEL, P.G. and KALETKA J. Advances in rotorcraft system identification, *Progress in Aerospace Science*, 1997, **33**, pp 259-284.
3. HAGAN, M.T, DEMUTH, H.B. and BEALE, M.H. *Neural Networks Design*, 1996, Boston, MA, USA, PWS Publishing.
4. CHUI, S.L. A cluster estimation method with extension to fuzzy model identification, Proceedings of the 3rd IEEE Conference on Fuzzy Systems, *IEEE World Congress on Computational Intelligence*, 26-29 June 1994, **2**, pp 1240-1245, Orlando, FL, USA.
5. TAKAGI, T. and SUGENO, M. Fuzzy identification of systems and its applications to modelling and control, *IEEE Transactions on Systems, Man and Cybernetics*, 1985, **15**, (1), pp 116-132.
6. JATEGAONKAR, R.V. *Flight vehicle system identification: a time domain methodology, Progress in Astronautics and Aeronautics Series*, Published by AIAA, ISBN-10: 1-56347-836-6, 2006, **216**, 1st ed.
7. MILLIKEN, W. F. Dynamic stability and control research, Proceedings of the 3rd Anglo-American Aeronautical Conference, Brighton, UK, 1951, pp 447-524.
8. DERUSSO, P, ROY, R.J., CLOSE, C.M. and DESROCHERS, A.A. *State Variables for Engineers*, 2nd ed, Wiley and Sons, New York, USA, 1998.
9. NELLES, O. *Nonlinear System Identification: From Classical Approaches To Neural Networks And Fuzzy Models*, ISBN-3-540-67369-5, Edition Springer-Verlag Berlin Heidelberg, New York, USA, 2001.
10. RYSDYK, R. and CALISE, A.J. Robust nonlinear adaptive flight control for consistent handling qualities, *IEEE Transactions on Control Systems Technology*, 2005, **13**, (6), pp 896-910.
11. TISCHLER, M.B. and KALETKA, J. Modelling XV-15 tilt-rotor aircraft dynamics by frequency and time-domain identification techniques, *AGARD: Rotorcraft Design for Operations*, p 20, 1987.
12. BOHLIN, T. *Practical grey-box process identification: theory and applications*, USA, London: Springer-Verlag London, 2006, (http://dx.doi.org/10.1007/1-84628-403-1).
13. NADEAU BEAULIEU, M. and BOTEZ, R.M. Simulation and prediction of the helicopter main rotor, tail rotor and engine parameters by using the subspace system identification method, *J Aerospace Eng*, 2008, **222**(G6), pp 817-834.
14. NADEAU BEAULIEU M., BOTEZ, R.M. and HILIUTA, A. Ground dynamics model validation by use of landing flight test, *AIAA J Aircraft*, 2007, **44**, (6), pp 2063-2068.
15. PROUTY, R.W. *Helicopter Performance, Stability, and Control*, Malabar, 2002, Flor., R. E. KRIEGER.
16. ELSHAFEI, M., AKHTAR, S. and AHMED, M.S. Parametric models for helicopter identification using ANN, *IEEE Transactions on Aerospace and Electronic Systems*, **36**, (4), pp 1242-1252.
17. SAMAL, M. K., ANAVATTI, S. and GARRATT, M. Neural network based system identification for autonomous flight of an Eagle Helicopter, Proceedings of the 17th World Congress of the International Federation of Automatic Control, 6-11 July 2008, Seoul, Korea.
18. MONTAZER, GH., A., SABZEVARI, R. and KHATIR, H., Gh. Improvement of learning algorithms for RBF neural networks in a helicopter sound identification system, *J Neurocomputing*, 2007, **71**, (1-3).
19. CABELL, R.H., FULLER, C.R. AND O'BRIEN, W.F. A neural network for the identification of measured helicopter noise, *J American Helicopter Society*, 1993, **38**, (3).