ONLINE PARAMETER ESTIMATION OF A MINIATURE UNMANNED HELICOPTER USING NEURAL NETWORK TECHNIQUES

Paulin Kantue

A dissertation submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in fulfillment of the requirements for the degree of Master of Science in Engineering.

Johannesburg, 2011

DECLARATION

I declare that this dissertation is my own unaided work. it is submitted for the Degree of Master of Science in Engineering to the University of Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination to any other University.

.....

..... day of 2011

Acknowledgments

This dissertation would not have been possible without God's presence in my life. I am especially thankful to my wife Lerato, my parents, my sister Rita and my friends for always encouraging me even when the future of my research was unclear. I am heartily thankful to my supervisor and personal mentor Dr. J.O. Pedro for the countless hours of help and guidance. His patience and support from the concept stage until now has enabled me to develop an understanding in the subject. I would like to make special reference to Mr. Ian Wagstaff of Advanced Technologies of Engineering. Without his support and corporation I could not have gotten to this point. Lastly, I offer my warm regards and blessings to all those who have encouraged and supported me during the completion of this dissertation.

Paulin Kantue

Abstract

The online aerodynamic parameter estimation of a miniature unmanned helicopter using Neural Network techniques has been presented. The simulation model for the miniature helicopter was developed using the MATLAB/ SIMULINK software tool. Three trim conditions were analyzed: hover flight, 10m/s forward flight and 20m/s forward flight. Radial Basis Function (RBF) online learning was achieved using a moving window algorithm which generated an input-output data set at each time step. RBF network online identification was achieved with good robustness to noise for all flight conditions. However, the presence of atmospheric turbulence and sensor noise had an adverse effect on network size and memory usage. The Delta Method (DM) and the Modified Delta Method (MDM) was investigated for the NN-based online estimation of aerodynamic parameters. An increasing number high confidence estimated parameters could be extracted using the MDM as the helicopter transitioned from hover to forward flight.

Published Works

The following conference papers associated with this dissertation have been published:

1. Pedro, J. O. & Kantue, P. (2010), 'Online Identification of a Miniature Unmanned Helicopter Using Radial Basis Functions, Proceedings of the International Aerospace Symposium of South Africa, Cape Town, South Africa
2. Pedro, J. O. & Kantue, P. (2011), 'Online Aerodynamic Parameter Estimation of a Miniature Unmanned Helicopter Using Radial Basis Function Neural Networks, Proceedings of the 8th Asian Control Conference, Kaohsiung, Taiwan, pp. 1170-1175

Contents

List of Figures

List of Tables

Nomenclature

ACRONYMS

ANN Artificial Neural Networks

BPA Backpropagation Algorithm

DM Delta method

EKF Extended Kalman Filtering

$FFNN$ Feed Forward Neural Network

GPS Global Positioning Sensor

GUI Graphical User Interface

IMU Inertial Measurement Unit

KF Kalman Filter

MDM Modified Delta method

MIT Massachusetts Institute of Technology

MLP Multi-layer Perceptron

NN Neural Network

OLS Orthogonal Least Square

PSD Power Spectral Density

$RBFN$ Radial Basis Function Network

RFC Reconfigurable Flight Control

$RMLP$ Recurrent MultiLayer Perceptron

RNN Recurrent Neural Network

TPP Tip Path Plane

UAV Unmanned Aerial Vehicle

$VTUAV$ Vertical Take-off and Landing Unmanned Aerial Vehicle

GREEK SYMBOLS

δ_r^{trim} Tail rotor trim offset

δ_{col} Collective control input

$\delta_{lat}, \delta_{lon}$ Lateral and longitudinal cyclic control inputs

δ_r Yaw pedal control input

δ_t Engine thrust level input

ϵ_{vf}^{tr} Fraction of vertical fin exposed to Tail Rotor induced velocity

γ_{fb} Stabilizer bar Lock number

$\in_{vf}^{tr}$ Fraction of vertical fin area exposed to Tail rotor induced velocity

μ Advance ratio

μ_z Normal airflow component

Ω_{nom} Nominal main rotor speed

Ω Rotor angular speed

ω_n Natural undamped frequency

ϕ, θ, ψ Rolling, pitching and yawing body angles

ξ^s Damping ratio of the suspension system material

ζ Damping ratio

LATIN SYMBOLS

$A_{\delta_{lon}}^{nom}$ Longitudinal cyclic to flap gain at nominal rpm

a_1, b_1 longitudinal and lateral main rotor flapping angles

a_x, a_y, a_z Body accelerations along the x, y and z axis

$A_{\delta_{lon}}$ Effective steady-state longitudinal gain from the cyclic input to the main rotor flap angles

a_{mr} Main rotor blade lift curve slope

a_{tr} Tail rotor lift curve slope

$B_{\delta_{lat}}^{nom}$ Lateral cyclic to flap gain at nominal rpm

$B_{\delta_{lat}}$ Effective steady-state lateral gain from the cyclic input to the main rotor flap angles

$C_{L_\alpha}^{ht}$ Horizontal tail lift curve slope

$C_{D_0}^{mr}$ Main rotor blade zero lift drag coefficient

$C_{T_{max}}^{mr}$ Main rotor max thrust coefficient

$C_{D_0}^{tr}$ Tail rotor blade zero lift drag coefficient

$C_{T_{max}}^{tr}$ Tail rotor max thrust coefficient

$C_{L_\alpha}^{vf}$ Vertical fin lift curve slope

c_{mr} Main rotor chord

c_{tr} Tail rotor chord

f_p^s Rolling resonance frequency of the suspension system

f_q^s Pitching resonance frequency of the suspension system

f_r^s Yawing resonance frequency of the suspension system

h_{mr} Main rotor hub height above c.g.

h_{tr} Tail rotor hub distance from c.g.

$I_{\beta_{mr}}$ Main rotor blade flapping inertia

I_{xx}, I_{yy}, I_{zz} Rolling, pitching and yawing moment of inertia

K_β Hub torsional stiffness

K_μ Scaling of flap response to speed variation

K_i Integral governor gain

K_p Proportional governor gain

l_{ht} Stabilizer location behind c.g.

L_{mr}, N_{mr}, M_{mr} Main rotor rolling, pitching and yawing moment

l_{tr} Tail rotor hub location behind c.g.

m Helicopter mass

n_{es} Gear ratio of engine shaft to main rotor

n_{tr} Gear ratio of tail rotor to main rotor

P_{eng}^{idle} Engine idle power

P_{eng}^{max} Engine max power

Q_e Engine torque

Q_{mr} Main rotor torque

Q_{tr} Tail rotor torque

R_{mr} Main rotor radius

R_{tr} Tail rotor radius

S_x^{fus} Frontal fuselage drag area

S_y^{fus} Side fuselage drag area

S_z^{fus} Vertical fuselage drag area

S_{ht} Horizontal fin area

S_{vf} Effective vertical fin area

$T_{1/2}$ Time to double or half amplitude

T_{mr} Main rotor thrust

u, v, w Body translational velocities

$u_{wind}, v_{wind}, w_{wind}$ Wind velocities along the x,y,z axis respectively

V_{imr} Main Rotor induced velocity

V_{itr} Tail rotor induced velocity

x_E, y_E, z_E Position of the body c.g w.r.t to the inertial reference system

Y^{tr} Tail rotor thrust

SUBSCRIPTS

0 Constant term

col collective input

E inertial reference

e engine

es engine shaft

imr Main rotor induced

itr Tail rotor induced

lat lateral cyclic input

lon longitudinal cyclic input

mr Main rotor

r yaw pedal input

t throttle lever input

tr Tail rotor

$wind$ wind direction

x, y, z Along the the x,y and z axis

SUPERSCRIPTS

fus Fuselage

ht Horizontal stabiliser

$idle$ Idle value

max Maximum value

nom Nominal value

vf Vertical fin

Chapter 1

Introduction

1.1 Research Background

There has been a steady growth in the development of rotorcraft unmanned aerial vehicles (UAVs) using reconfigurable flight control (RFC) systems (Kumar et al., 2003). The global idea behind RFC is that after a failure has occurred, the flight control system is adapted such that the overall system characteristics minimize or mitigate further closed loop system deterioration. This is usually achieved through the online alteration of the control system's characteristics in the event the aircraft sustains flight control system performance degradation.

These adaptive systems have greatly improved the reliability in robotics operations which include: surveillance, disaster assistance, search and rescue (civilian and military), data and image acquisition of targets/areas, electronic attack, strike missions, suppression and/or destruction of enemy air defense. Boeing A160 Hummingbird is an example of an unmanned helicopter (see Figure.1.1) which integrates these new technologies allowing for greater endurance and altitude than any other helicopter in operation. It has a maximum speed of 258 km/h, endurance of over 20 hours and a service ceiling of 9150m.

However, human factors have been consistently identified as a major cause of unmanned aircraft accidents (Williams, 2004; Williams, 2006). The percentage of accidents attributed to human error ranges between 70 to 80 percent. Understanding these factors is important to improve the reliability of these aircraft to a level comparable to manned aircraft. The development of reconfigurable flight control systems with online parameter estimation techniques is one way to improve the reliability of unmanned aircraft (de Weerdt, 2005).

A good example proving the desirability of a reconfigurable flight control system in rotorcraft , is the accident of the Navy-owned Vertical Take-off and Landing Tactical Unmanned Aerial Vehicle (VTUAV), Fire Scout (see Figure.1.2). The investigation conducted on the 4th of November 2000 revealed that human error combined with a damage of the onboard anten-

Figure 1.1: Boeing A160 Hummingbird UAV.

nas during ground handling, led to the accident. The damaged antennas emitted an incorrect signal causing the radar altimeter to incorrectly track the altitude. The antennas issued a false reading which indicated that the Fire Scout was at an altitude of 2ft above the ground when, in fact, it was hovering at an altitude of 500ft. After the 'land' command was given, the guidance and control system interpreted the incorrect altitude signal indicating that the Fire Scout was landing. As in accordance with flight procedure, the command was given to shut down the engine which caused the aircraft to crash. It was concluded that a unique approach to automation and procedures often results in unforeseen and costly outcomes (Williams, 2004).

Figure 1.2: Navy Fire Scout in flight.

1.2 Research Rationale and Motivation

A neural network-based approach to the system identification of miniature helicopters has shown to facilitate the development of accurate and flight-validated model (Samal et al., 2008). As neural networks do not require to estimate initial values for the parameterized

model, the helicopter stability derivatives can be directly computed from flight or simulated data.

This research study could have a potential impact on reducing development cost for applications such as: development of aerodynamic databases for training simulators and controller design. The parallel nature and fast adaptability of neural networks are well suited for controller hardware design and implementation for miniature helicopters.

1.3 Problem Statement

The rigid rotorhead of miniature helicopters permits for large rotor forces and moments which enables them to perform sustained inverted flight. For many years, these systems have been described using linearized models of the various steady state flight conditions such as hover and forward flight. This approach is insufficient as a higher order model is required to completely describe the dynamics of this special class vehicle. Moreover, the exact order and structure of the mathematical model is needed to apply parametric estimation techniques such as Maximum likelihood and Kalman filter. An alternative approach is required for the design of these unmanned systems, fully exploiting the capabilities presented by their complex dynamic behaviour.

A neural network-based online aerodynamic parameter estimation methodology is proposed. Neural networks do not require a priori information of the system. As general function approximators, they are able to identify the underlying dynamics of an unstable, nonlinear and uncertain system within a noisy environment using flight data. The online application of this method will take us a step further in answering whether the design of a flight reconfigurable unmanned systems with the full capability of miniature helicopter can be achieved.

1.4 Literature Review

Miniature helicopters have been used extensively in the development of unmanned aerial vehicles. Their inherent ability to take-off and land vertically, hover, forward flight and aerobatic maneuvers have made them ideal for flight through confined spaces such as urban areas. The Yamaha R-50 (RTMAx) is a commercial model-scale helicopter which has become a popular platform for many researchers (Mettler et al., 1999; Mettler, Kanade and Tischler, 2000; Mettler, Kanade, Tischler and Messner, 2000). This aircraft was originally designed for crop-dusting applications, but was later modified by Carnegie Mellon Robotics Institute for vision-based autonomous flight (Mettler et al., 1999).

Miniature helicopters have a higher thrust-to-weight ratio than their full-scale counterparts (Gavrilets et al., 2003). Firstly, as the vehicle size decreases, the moments of inertia decrease with the fifth power of the scaling factor. Secondly, the rotor head of a miniature helicopter is

relatively stiffer allowing for large rotor forces and moments. Many of these helicopters can produce negative thrust enabling sustained inverted flight and making miniature helicopters outperform most agile full-scale helicopters (Gavrilets et al., 2003). The complex interactions taking place between the rotor wake, fuselage and tail in full-scale helicopters are negligible in comparison to the large rotor forces and moments generated from the rotor control inputs in miniature helicopters. Gavrilets (2003) concluded that applying detailed first-principle modelling techniques (Prouty, 1986; Padfield, 1996) on miniature helicopters is inadequate as there is a dramatic increase in agility as the flight vehicle size reduces.

Modelling techniques based on system identification have been used to derive linear models (Mettler et al., 1999; Mettler, Kanade and Tischler, 2000; Ljung, 1997), control system design (Mettler, Kanade, Tischler and Messner, 2000), aerodynamic parameter identification/estimation (McNally and Bach Jr., 1988) and validation of detailed non-linear first-principle models (Tischler, 1995). Frequency domain identification methods such as CIFER (Mettler et al., 1999; Mettler, Kanade and Tischler, 2000; Mettler, Kanade, Tischler and Messner, 2000; Tischler, 1995) have been used to develop accurate parameterized models around specific trim conditions. In the last twenty years, methods to generate real-flight and simulated data have been used extensively for aerodynamic parameters estimation (McNally and Bach Jr., 1988). Within the rotorcraft system identification framework, these modelling techniques limit the operation of model-scale helicopter UAV to hover and low-speed forward flight as the inherent state space modelling approach does not account for powered flight dynamics and possible in-flight reconfiguration (Mettler, Kanade and Tischler, 2000). Kim and Tilbury (2004) have attempted to solve this problem by including fly-bar dynamics in the mathematical model of the Ikarus ECO unmanned helicopter.

1.4.1 Online aerodynamic parameter estimation methods

Conventional methods

As early as the 1970s, maximum likelihood methods have been largely used for parameter estimation problems. In this approach, the stability and control parameters are assumed and then used in the aerodynamic model to validate the dynamic model (Samal et al., 2008). Moreover, maximum likelihood methods proved they can handle both process and measurement noises. They can also be used to estimate noise covariance which eliminates the problem of specifying a weighting matrix for the covariance of measurement noise errors (Mehra et al., 1974). A maximum likelihood method using Gauss-Newton algorithm (Jategaonkar, 2000) was used to develop a MATLAB-based parameter identification toolbox for the miniature unmanned helicopter ARTIS at DLR (Lorenz and Chowdhary, 2005).

Output error methods can be considered as maximum likelihood methods where process noise is not present. It is arguably the most widely applied methods to estimate aerodynamic derivatives from flight data. These methods can be applied to nonlinear systems with arbitrary complexity, although the presence of atmospheric turbulence often yields biased results

(Jategaonkar, 2008). Moreover, equation error methods can be seen as maximum likelihood methods where no measurement noise is present and all the states are measured (Mehra et al., 1974). These methods have been used for model structure development and parameter estimation of a F/A-18E Super Hornet (Paris and Bonner, 2004).

Kalman filtering is also a well known technique for state and parameter estimation. It is a recursive estimation procedure using measurement data sets in sequence. The assumed states are improved at each time step by taking prior states estimates and new data from succeeding state estimation. This approach was used with a sensor model to estimate the states of the USC AVATAR miniature unmanned helicopter (Jun et al., 1998).

Morelli (2000) made use of frequency-domain equation error method for the real-time parameter estimation from flight test data of a F-18 High Alpha Research Vehicle (HARV). A recursive Fourier transform algorithm applied to a linear dynamic state-space model was used for real-time data analysis. The presence of high Gaussian random white noise and simulated data dropout, which represented high frequency noise, did not prevent the parameter estimates to converge to their true value. Lack of *a priori* information increased the convergence time, which made this method inadequate for adaptive or reconfigurable flight control application.

Velo and Walker (1997) presented a note on the aerodynamic parameter estimation of high performance aircraft using the Extended Kalman Filtering method. Values for the process and measurement noise covariance were assumed to produce estimates of the states and a predicted state estimation error covariance. The white noise process of intensity, also known as pseudo-noise, was used to account for modelling errors by allowing the parameter time variation to compensate for unmodelled dynamics. This allows faster convergence of the parameter estimates at the expense of increased parameter error variance by keeping the filter gains high.

de Weerdt (2005) used an Extended Kalman Filter algorithm for the aerodynamic parameter estimation of a F-16 aircraft. The state values were estimated and at the same time the real values of the measurement biases were computed. This method uses the linearized state-space model about the most recent state estimate at each time step, allowing for the sequential updating of model parameters. Difficulties were found in the estimation of the measurement biases and rotational accelerations. However, the estimation of the aerodynamic model was based on the interpolation of low-speed static and dynamic wind tunnel test results.

Song et al. (2001) compared two online parameter estimation techniques for a fault tolerant flight control system application. It was observed that conventional least square regression methods such as: Recursive least square (RLS), RLS with a forgetting factor, Modified Sequential Least Square, real-time Batch Least squares and the extended Kalman filtering methods lacked parameter reliability in the presence of unmodelled noise and cross-coupled control inputs (Lombaerts et al., 2009)

Neural network-based methods

The exact order and structure of a mathematical model might not be known to apply parameter estimation techniques. Since helicopter is a complex, highly coupled and non-linear system, it needs a higher-order method to completely describe the system. In addition, measurement noise generated by sensors and structural vibration adds to the system identification problem. The non-conventional neural network-based methods are an alternative non-parametric approach that has gained popularity in high-performance autopilot, control systems and parameter estimation (White and Sofge, 1992; Demuth and Beale, 2002).

Neural networks are broadly classified on the basis of the type of connectivity between the processing elements (neurons), the type of architecture, and the number of layers in the network. Two types namely: recurrent neural networks (RNNs) and feed-forward neural networks (FFNNs) have been extensively applied to the aerodynamic parameter identification of non-linear systems (Songwu and Basar, 1998; Kumar et al., 2008). RNNs are dynamic neural networks incorporating output feedback. They appear in different forms such as: Non-linear Auto Regressive eXogenous (NARX) input model, ANN with internal memory known as Memory Neuron Networks and Recurrent MultiLayer Perceptron (RMLP) networks. A comparative study was done on these networks for the identification of helicopter dynamics from flight data (Kumar et al., 2003).

FFNNs are static in nature. They work as function approximators and are capable of approximating any continuous function to any degree of accuracy. However, this leads to a black-box model where there is no tangible relation to the physical parameters that can be attached to the model structure or the computed weights (Raisinghani, Ghosh, and Kalra, 1998). The radial basis function is an alternative to the nonlinear parameter neural networks as the network weights are adjusted using the Least squares method. It can be regarded as a special two-layer FFNN which is linear in the parameters by fixing all the RBF centers and nonlinearities in the hidden layer. RBF networks can result in poor estimations unless a systematic approach to center selection is undertaken (Chen et al., 1991). RBF networks have been extensively used in numerous parameter estimation studies (Raisinghani, Ghosh and Khubchandani, 1998; Raisinghani and Ghosh, 2001; Singh and Ghosh, 2007; Song et al., 2005). The multi-layer perpectron (MLP) is the more popular type of FFNNs used for parameter estimation, although it has some drawbacks such as: slow convergence rate, computational memory and sensitivity to outliers (Kumar et al., 2008).

Raisinghani, Ghosh, and Kalra (1998) applied the delta and zero method for aircraft parameter estimation using FFNNs from simulated and real-flight data. The delta method (DM) is based on the understanding that a stability derivative is the variation of the aerodynamic force or moment caused by a small variation in one of the motion/control variables about the nominal value, while other variables remain constant (Raisinghani, Ghosh and Khubchandani, 1998). In contrast, the zero method is based on the ratio of aerodynamic coefficient to the motion/control variable, due to a change of motion/control variable while all the other variables are set to zero. Specific use of these methods with RBFNs (Kumar et al., 2008), and

within a frequency domain context, has been explored (Raisinghani and Ghosh, 2001). Minimization of the output layer error was done using a back propagation algorithm (BPA) with the gradient descent method.

However, these methods had yielded different estimates at different time points making the delta method less than perfect. Moreover, large standard deviations in the estimated parameters, caused the results to loose credibility (Raisinghani and Ghosh, 2001). To rectify this problem, the 'modified' delta method has been applied for estimation of lateral-directional parameters (Singh and Ghosh, 2007). This is very similar to the delta method except that the mapping of input-output variations is considered. This method was applied on the DLR fixed-wing testing aircraft system (Singh and Ghosh, 2007).

Songwu and Basar (1998) investigated the problem of driving noise using FFNN and RBF network models based on H_∞ identification algorithm. It was found that this scheme and the genetic algorithms-based scheme outperformed the backpropagation algorithm in terms of speed of convergence. It was concluded that RBFNs were better suited for low-dimensional systems as it has fewer parameters to identify.

Later that year, the delta and zero methods were again used by Raisinghani, Ghosh and Khubchandani (1998), for the estimation of aircraft stability and control derivatives. However the presented results had the cases whereby multistep 3-2-1-1 input, arbitrarily varying input, sinusoidal input and a variation of these for both aileron and rudder was introduced. Robustness of the methods with respect to measurement noise was demonstrated through its application on both simulated and flight data. It was deduced that the delta method could be used as an alternative and complementary to existing parameter estimation methods such as the maximum likelihood methods.

Raisinghani and Ghosh (2001) developed the delta method for frequency domain application using a discrete Fourier Frequency Transform (FFT) with real flight data. The frequency transform output variable was split into real and imaginary parts. Then, the delta method was performed on both parts separately and plotted at each frequency. The mean value of the normal distribution was taken as the estimated value and the sample standard deviation about the mean as the measure of accuracy of the estimates. The results produced much lower standard deviation of estimates and correlated well with literature. This methodology proved very effective in eliminating high frequency noise while retaining signal content essentially intact.

Singh and Ghosh (2007) applied the modified delta method (MDM) in aircraft parameter estimation. The neural network was trained using differential variation of aircraft motion/control variables as inputs and coefficients as outputs. This method was validated using both simulated and real flight data. Modified back-propagation algorithm was introduced as the traditional approach leads to extremely slow convergence for very small values and large values often lead to parasitic oscillations. This method yielded estimates with lesser standard deviations and unlike popular estimation algorithms, it did not require order of magnitude information for the parameters.

Kumar et al. (2008) implemented the integration of the delta method and feedforward neural networks for a full-scale helicopter. This was applied on a RBF and MLP network. The simulated data was generated using a 6-DOF nonlinear simulation model. The modified 3211 pilot control input was used for rotorcraft parameter estimation. The RBFN based delta method was found to be suitable for rotorcraft parameter estimation as it enables the computation of aerodynamic derivatives in both the transition regime and high speed flight regime.

Suresh et al. (1995) worked on the identification of lateral and longitudinal dynamics of a helicopter using recurrent neural networks. FFNNs with linear filters also known as the Narendra's model and RNNs with internal memory (MNN) was used. The training and testing error for MNN was far less than the Narendra's model which has more number of weights due to delay lines. It was concluded that fine-tuning of the network parameters can further enhance the performance of the networks.

A comparative study into the identification of helicopter dynamics based on flight data using recurrent neural networks was presented in (Kumar et al., 2003). The NARX, MNN and RMLP networks were used to identify longitudinal and lateral dynamics of the helicopter at various speeds. The research platform was a four-bladed soft in-plane hingeless main rotor and a four bladed tail rotor with conventional mechanical controls. These networks could be trained on-line and off-line but their application would depend on *a priori* knowledge and amount of training data available.

de Weerdt (2005) investigated the problem of reconfigurable flight control system using online neural network-based parameter estimation techniques. The adaptive nature of the aerodynamic model was based on the assumption that thrust and mass distribution data was available at all times. A hyperbox structure for the whole flight envelope was defined to resolve the recency effect which occurs during online identification whereby the neural network is unable to retain the approximation accuracy of the previous Input-Output mappings. However, this led to the construction of many neural networks in order to completely describe the full-envelope aerodynamic model. Off-line learning of a longitudinal aerodynamic model was performed in order to validate the proposed RFC system.

Savran et al. (2006) developed a neural network-based adaptive identification model for three angular rates of a high performance aircraft. This model made use of a first-in first-out stack to store certain history of the input-output data to be used for offline and online training of the neural networks. The convergence time was reduced by using Levenberg-Marquardt optimization method with a trust region approach. It was concluded that the stack method enhanced the adaptive neural network capabilities in compensating for system uncertainties and adapt to parameter changes in real time.

The online identification using adaptive RBF-based neural networks was investigated by Jafari et al. (2007). They applied a modified growing and pruning radial basis functions (GAP-RBF) and the minimal resource allocation network (MRAN) for the online identification of nonlinear systems. The Unscented Kalman Filter (UKF), with a variable forgetting factor, was used as a learning algorithm to update the neural network parameters. Unlike

batch training, the series of training samples, were presented one-by-one to the network. Although these methods update the network parameters in real-time, their networks initially start at a neuron count of zero which leads to big errors in the identification results and slow convergence time.

1.4.2 Flight control reconfiguration using neural network

Leitner et al. (1995) conducted a study bridging the gap between adaptive neural network and helicopter nonlinear modelling. Emphasis was on the network architecture and the effect varying adaptation gain had on the control system tracking capability. The analysis of the feedback inversion error was used to decide the network architecture and the basis functions. A proportional plus derivative (PD) control law was used to shape the response and the adaptive signal sole purpose was to cancel the inversion error. The approximate nature of the inverse method required the neural network model to be developed using off-line training. So high adaptation gains resulted as a trade-off for short control activity although the *spill-over* effect due to actuator saturation was not quantified. Pallett and Ahmad (1991) investigated real-time flight control of a miniature helicopter using neural networks. The identification strategy made use of the multilayer perceptron (MLP) networks for the offline training of the inverse dynamics. This is done so that network initial weights are close to the operating point of the system. The control design techniques were based on linearized state equations in the hover and vertical flight conditions.

Chuntao and Yonghong (2006) investigated the hysteresis characteristics of piezoelectric actuators in relation to the adaptive control of neural networks. Construction of the complex hysteresis inversion model was avoided by using the backstepping technique and backlash inversion to compensate the backlash nonlinearity. It was assumed the actuator output and all states of plant are measurable which is seldom achieved in the identification and control of model helicopters. Johnson and Kannan (2005) had a tracking objective but this was handled by an inner-outer loop control architecture approach. Their autonomous unmanned helicopter model made use of hedging to prevent outerloop adaptation of inner loop dynamics and avoid incorrect adaptation while at the control limits.

Calise and Rysdyk (1997) investigated nonlinear adaptive control using neural networks and the model inversion method. However, cross-coupling between fast rotational states and slow translational states was excluded in the inversion. The objective was to demonstrate how neural networks are capable of adapting to errors caused by the linearized inversion model. The control law was defined through the neural network weight adaptation which guaranteed the boundedness of the tracking error. The same method was used by Munzinger (1998) in the development of a real-time flight simulator for an experimental model helicopter.

Suresh and Kannan (2008) developed a direct adaptive controller design using neural networks for an unstable unmanned research aircraft. A neural network with linear filters and

back propagation through time learning algorithm was used. The bounded input-output requirement is overcome through the use of an offline-online training strategy. The adaptive nature of the neural controller was tested using center of gravity variation, system matrix uncertainty and control surface loss. Nonlinear simulations were conducted to analyze the robustness of the proposed control scheme. It was found that control effort required in direct adaptive scheme is lesser than the indirect adaptive control law.

Recently, Guo et al. (2010) presented an active fault accommodation strategy in the presence of actuator fault and input constraints. A combination of a direct adaptive control algorithm with multiple model switching was used to construct the reconfigurable flight controller. An adaptive observer to reconfigure the plant is designed using RBFN. The RBF were used to approximate the model uncertainty and the multiple models describe all the fault scenarios.

1.4.3 Identified gaps

- Least Square methods are the most popular methods for online parameter estimation and flight control. Although they inherently improve the signal-to-noise ratio, provided the noise covariance parameters are given, they require accurate *a priori* knowledge of the system dynamics and initial system parameters which is very difficult to get for miniature unmanned helicopters.
- MLP and RNN networks have been used extensively for the parameter estimation of the stability and control derivatives of nonlinear dynamic systems. Unlike RBF networks, these methods have slow convergence time and need large amount of required training data. This subsequently complicates their application to online estimation problems.
- The modified delta method has not been used for the parameter estimation of a miniature unmanned helicopter. Specific focus has been placed on a correlation between neural network performance to system identification accuracy.

1.5 Research Question

Can the online application of a neural network-based aerodynamic parameter estimation method be achieved for the design of a flight reconfigurable unmanned system?

1.6 Research Objectives

The aim of this study is to develop a simulation model that can accurately estimate aerodynamic parameters of miniature unmanned helicopter in real-time for flight control reconfig-

uration in the presence of state and measurement noise. The following sub-objectives should be noted:

- Development of a miniature unmanned helicopter simulation model.
- Evaluation of the trim and stability characteristics for a miniature unmanned helicopter.
- Evaluation of Neural Network architectures for nonlinear model identification.
- Online aerodynamic parameter estimation (stability and control derivatives) for a miniature unmanned helicopter using radial basis function networks (RBFN) and multilayer perceptron networks (MLPN).
- NN model structure selection for flight control reconfiguration.
- Evaluation of the NN-based control methodologies for reconfigurable flight control design.
- Evaluation of the robustness and performance of a miniature helicopter reconfigurable flight control system.

1.7 Research Scope and Limitations

In this research, the online aerodynamic parameter estimation of a miniature unmanned helicopter using neural network techniques is studied. A simulation facility for a miniature unmanned helicopter is developed using the MATLAB/SIMULINK software suite. The online estimation of the stability and control derivatives using RBFN-based DM and MDM is achieved for three flight conditions namely: hover, forward 10m/s flight and forward 20m/s flight. A structure of a NN-based reconfigurable flight control system is also proposed.

Due to time and financial constraints, the following limitations applies to this research:

1. Only the MATLAB/SIMULINK software suite was used for all computations.
2. Real-time flight data for a miniature unmanned helicopter was not used.
3. The flapping rates $\dot{a_1}, \dot{b_1}$ were not estimated but were part of the simulated data used for parameter estimation.
4. MLP model identification and aerodynamic parameter estimation was not performed.
5. RBF spread constant optimization was not investigated.
6. Only the mean squared error validation criterion was used. one-step ahead (OSA) and model predictive outputs criteria were discarded.
7. Unsteady flight and acrobatic manoeuvers were not considered.

1.8 Research Methodology

The research methodology outline is shown in Figure 1.3. A miniature unmanned helicopter is the proposed platform for this study. The dynamic model of a X-Cell .60 unmanned helicopter developed by Gavrilets et al. (2003) is used to develop the mathematical model in MATLAB/SIMULINK environment. The blade element theory is used to derive the nonlinear equations of motion. A simulation database containing inputs and outputs time histories will be developed for both hover and forward flight conditions.

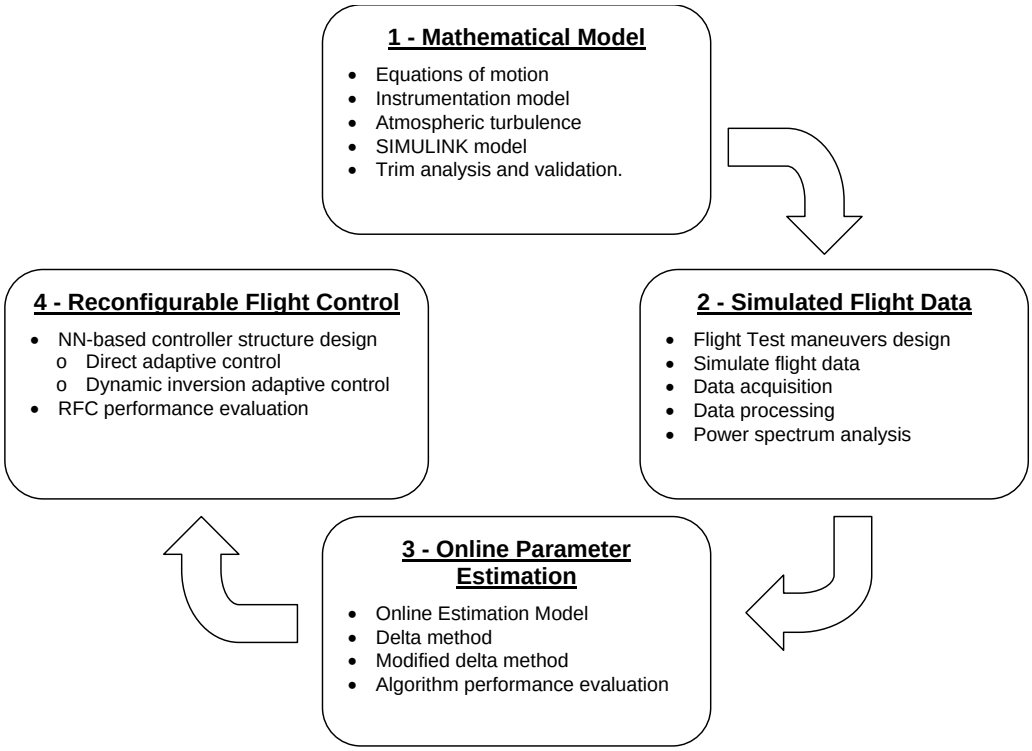

Figure 1.3: Methodology outline.

Instrumentation model

It is assumed that the sensor data is contaminated due to the vibration transferred by the airframe to the sensors. A measurement noise model will be developed which includes process and measurement noise at the inputs and outputs. A Gaussian noise model has been considered although specific focus on gyros low-frequency noise will be implemented.

Atmospheric turbulence model

Miniature helicopters are susceptible to atmospheric turbulence during flight. This effect is introduced through random signals from a white noise generator passing through a first-order filter to simulate various strengths of atmospheric turbulence.

Data acquisition

Simulated data should include enough information on the system dynamics to facilitate the system identification process. Specific maneuvers have to be designed to expose the response to inputs and the coupling effects. Lorenz and Chowdhary (2005) classified the maneuvers in the following categories:

- Varying 3-2-1-1 and doublet sweeps on individual inputs: To expose the effect of input/output relation.
- Combined varying 3-2-1-1 and doublet sweeps on two or more inputs: To expose the input coupling effects.

Data processing

The data files are expected to contain the following simulated parameters: pitch rate (q), pitch attitude (θ), roll rate (p), roll attitude (ϕ), yaw rate (r), longitudinal (a_x), lateral (a_y) and normal accelerations (a_z). The inputs to the servo actuators i.e collective (δ_{col}), aileron (δ_{lat}), elevator (δ_{lon}), and tail rotor pitch (δ_{ped}) will also be logged.

Parameter estimation algorithms

The following indirect ways of extracting stability and control derivatives from a trained network will be applied for parameter estimation:

- Delta Method.
- Modified Delta method.

The parameter estimation algorithm will be evaluated using the following criteria (Song et al., 2001):

- Convergence error of estimated parameters.
- Confidence levels of the estimated parameters through standard deviation computation.
- Network neuron size.
- Network memory usage.
- Robustness to process and measurement noise.

Reconfigurable flight control

The following NN-based control methodologies have been investigated:

1. Direct adaptive control
2. Dynamic inversion adaptive control

The neural network reconfigurable flight control law will be based on the estimated aerodynamic parameters. Attitude control will form the basis of the validation process with the following specifications:

1. Eigenvalue location
2. Gain and phase margins
3. Crossover frequency

1.9 Research Contributions

This study introduces a novel approach to the online aerodynamic parameter estimation of a miniature unmanned helicopter using neural network-based techniques. The following contributions have been identified:

1. Development and validation of a Xcell.60 miniature helicopter simulation facility for system identification applications.
2. Online application of RBFN to the aerodynamic parameter estimation of a miniature unmanned helicopter in hover and forward flight.
3. The application and comparison of the DM and MDM using RBFN for a miniature unmanned helicopter.

1.10 Dissertation Outline

The layout of this dissertation is as follows: Chapter 2 describes the development of a miniature helicopter simulation model. Chapter 3 analyzes the trim and stability condition of the helicopter in various flight phases with emphasis on natural modes of motion. Chapter 4 describes the online identification and aerodynamic parameter estimation using neural networks. Chapter 5 describes flight control reconfiguration using neural networks. Conclusions and recommendations for this research are given in Chapter 6.

Chapter 2

Development of a Miniature Unmanned Helicopter Model

2.1 Description of X-Cell.60 Helicopter

2.1.1 Airframe characteristics

The X-Cell.60, shown in Figure 2.1 is a very popular helicopter among hobby pilots for aerobatics and recently for research in highly maneuverable autonomous flight (Mettler et al., 1999). The X-Cell.60 is manufactured by Miniature Aircraft with a rigid hingeless rotor head. This allows for large rotor control moments resulting in high roll and pitch angular rates. In comparison to the more popular Yamaha R-50 (RTMax), the X-Cell has a larger thrust-to-weight ratio permitting fast acceleration, high load factors and sustained inverted flight.

The Bell-Hiller stabilizer bar is a secondary rotor consisting of two paddles mounted at each end of a steel rod and connected to the rotor shaft by a swashplate mechanism (see Figure 2.2). The bar receives the same cyclic input as the main rotor but is less sensitive to airspeed

Figure 2.1: Miniature X-Cell.60 helicopter.

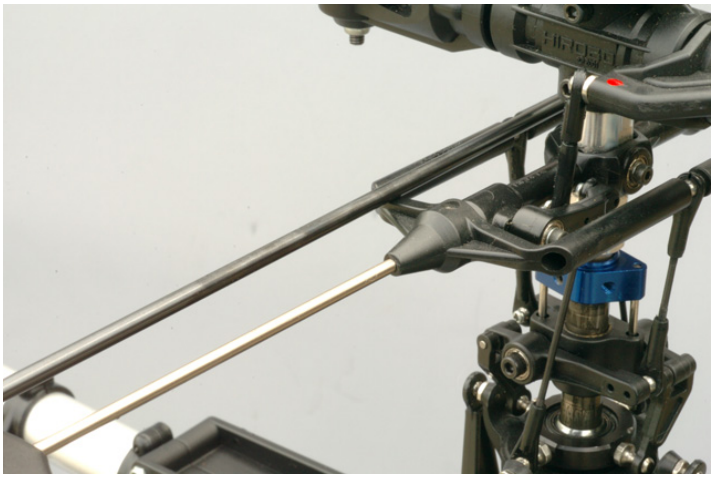

Figure 2.2: Helicopter stabilizer bar

and wind gusts due to a smaller blade Lock number (non-dimensional rotor parameter giving ratio between aerodynamic and inertial forces).The bar acts like a lagged attitude rate feedback (Prouty, 1986), designed to help the pilot control the helicopter attitude dynamics.

Swashplate mechanism, shown in Figure 2.3, is used as blade pitch control system. Its purpose is to vary the pitch of the blade both in magnitude and as a function of azimuth angle (angular position around the rotor hub). Unlike large helicopter which uses an intermediate mechanical mixing system, the X-Cell.60 helicopter makes use of the CCPM (Collective and Cyclic Pitch Mixing) to reduce the mechanical complexity and servo workload. The CCPM mixes the collective, cyclic longitudinal and cyclic lateral control inputs using software with three independent servos directly connected to the swashplate arranged in a 90 or 120 degrees design (Mettler, 2003).

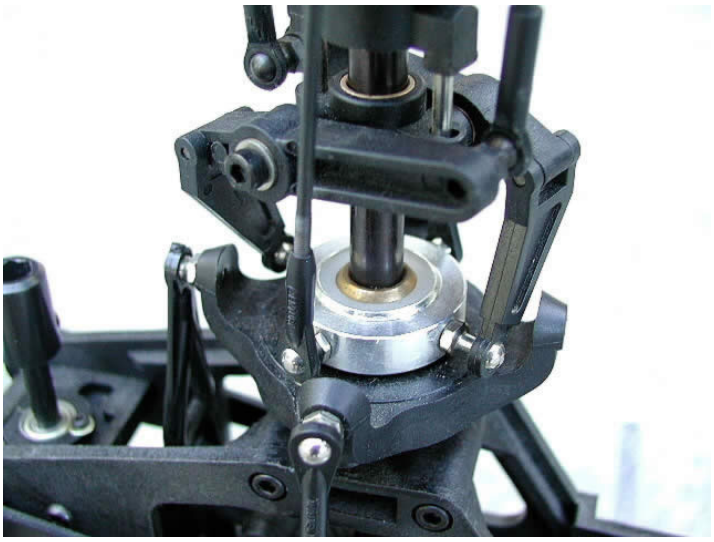

Figure 2.3: Helicopter swashplate mechanism

Most miniature helicopters use a heading lock gyro (see Figure 2.4). This acts as a yaw damping system using the negative feedback of the helicopter heading rate, provided by an angular rate gyroscope (gyro). Its purpose is to counter wind and main rotor reaction torque.

Using the gyro capabilities to 'sense' self-induced tail movements, a signal is generated to the actuator which in turn changes the tail rotor collective pitch to maintain constant heading (Day, 2005).

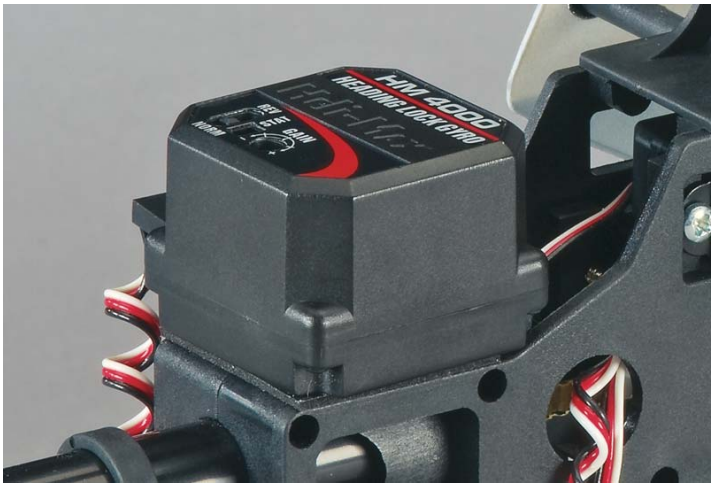

Figure 2.4: Helicopter Yaw Gyro

The X-Cell.60 is also equipped with a powerful .90 size engine and an electronic governor or speed controller whose function is to maintain constant rotor speed. Table 2.1 gives general characteristics of a X-Cell helicopter.

Table 2.1: Physical Characteristics of the X-Cell.60

Rotor speed	1600-1700 rpm
Tip speed	127-135 m/s
Dry weight	4.5 kg
Instrumentation	7.6 kg
Engine	2-stroke air cooled
Flight Autonomy	12 minutes

2.1.2 Instrumentation

Equipping a miniature aerobatic helicopter with multiple sensors puts hard requirements on the computing system. The measurements must be integrated and collected in real-time to accurately and directly measure the rapidly changing vehicle states. Instrumentation rigs fastened to the helicopter often limit the degrees of freedom and restrict the helicopter to hover flight experimentation (Mettler, 2003).

The development of the flight test instrumentation on the X-Cell.60 was done by Massachusetts Institute of Technology (MIT) (Gavrilets et al., 2003). A low-latency, high update rate unit GPS receiver was used, improving the accuracy of the attitude and velocity estimates using an onboard Extended Kalman Filter (EKF). The avionics suite is comprised of an inertial measurement unit which consists of three gyros and three accelerometers which supplies data at 50 Hz.

2.2 Rotorcraft Modelling

Rotorcraft modelling often falls within two categories: first-principles modelling and modelling using system identification. The former approach is a comprehensive analysis of all the vehicle's physical features. This makes the model suitable for a wide range of operating conditions but typically results in a large number of states. The latter approach focuses on developing compact and easier-to-understand models that capture the essence of the plant dynamics. This is achieved by estimating the system's responses from flight data collected during flight and ground experiments (Mettler, 2003).

The modelling approach is based on a study done by Mettler (Mettler, 2003) on the characteristics of small-scale helicopter dynamics using identification techniques. He applied identified parameterized linear models for both hover and forward flight on MIT's X-Cell.60 (Gavrilets et al., 2003). These models neglected complex interactions between the rotor wake, fuselage and tail due to the large rotor forces and moments produced in miniature helicopters.

Both hover and forward flight conditions are accurately modeled by a rigid-body model augmented with a first-order rotor and stabilizer bar dynamics with the exclusion of inflow dynamics. The lateral and longitudinal tip-path-plane (TPP) flapping equations were defined by lumping the coupled rotor and stabilizer bar equations into one first-order effective rotor equation. This allows the model to accurately predict the vehicle angular response to aggressive control inputs for the full range of the angular motion. A detailed study of the identification process used on the Yamaha R-50 and the X-Cell.60 can be found in (Mettler, 2003).

The goal of the mathematical model is to develop a simulation facility valid over the hover, slow speed (10 m/s) and medium speed (20 m/s) forward flight conditions using the MATLAB/SIMULINK software suite. A modular approach was undertaken to construct the aerodynamic model, sensor models, actuator models and atmospheric model. The helicopter parameters, derived from experimental tests, are listed in Table 2.2 (Mettler, 2003).

2.2.1 Assumptions

The rotorcraft model was developed using the following assumptions (Gavrilets et al., 2003):

1. The cross-axis moments of inertia are small and the principal axes coincide with the axes of the body reference system.
2. The fuselage center of pressure coincides with the c.g., therefore the moments created by three fuselage aerodynamic forces are neglected.
3. X-Cell cyclic control authority is dominated by the hub torsional stiffness making the modelling of the inflow transients less critical.

Table 2.2: Parameters of MIT Instrumented X-Cell.60 Helicopter

Parameter	Value	Parameter	Value
$A^{nom}_{\delta_{lon}}$	4.2 rad/rad	I_{yy}	0.34 $kg.m^2$
a_{mr}	5.5 rad^{-1}	I_{zz}	0.28 $kg.m^2$
a_{tr}	5.0 rad^{-1}	K_β	54 $N.m/rad$
$B^{nom}_{\delta_{lat}}$	4.2 rad/rad	K_i	0.02 rad^{-1}
$C^{vf}_{L_\alpha}$	2.0 rad^{-1}	K_p	0.01 sec/rad
$C^{ht}_{L_\alpha}$	3.0 rad^{-1}	K_μ	0.2
$C^{mr}_{D_0}$	0.024	l_{ht}	0.71 m
$C^{mr}_{T_{max}}$	0.0055	l_{tr}	0.91 m
c_{mr}	0.058 m	m	8.2 kg
$C^{tr}_{D_0}$	0.024	n_{es}	9.0
$C^{tr}_{T_{max}}$	0.05	n_{tr}	4.66
c_{tr}	0.029 m	Ω_{nom}	167 rad/s
δ^{trim}_r	0.1 rad	P^{idle}_{eng}	0.0 $Watts$
ϵ^{tr}_{vf}	0.2	P^{max}_{eng}	2000.0 $Watts$
f^s_p	12.5 Hz	R_{mr}	0.775 m
f^s_q	9.0 Hz	R_{tr}	0.13 m
f^s_r	9.6 Hz	S^{fus}_x	0.1 m^2
γ_{fb}	0.8	S^{fus}_y	0.22 m^2
h_{mr}	0.235 m	S^{fus}_z	0.15 m^2
h_{tr}	0.08 m	S_{ht}	0.01 m^2
$I_{\beta_{mr}}$	0.038 $kg.m^2$	S_{vf}	0.012 m^2
I_{xx}	0.18 $kg.m^2$	ξ^s	0.05

4. The influence of the cyclic input and roll rate on thrust are of second order for an advance ratio of $\mu < 0.15$ are neglected.

5. At hover, the vertical acceleration was represented by a linear function.

6. The rotor in-plane force, which contributes to the drag and side force was lumped with the fuselage forces.

7. The main rotor flapping angle β is represented as a Fourier series of the blade azimuth angle ψ, with the first three coefficient retained.

8. The main rotor and stabilizer bar flapping dynamics were lumped and represented by the TPP flapping dynamics with only two states.

9. The pitch and roll cross-coupling flapping coefficients were neglected.

10. The contribution of the rotor moments is approximated using a linear torsional spring with constant stiffness.

11. The main rotor thrust vector remains perpendicular to the TPP.

12. Small flapping angles are considered, thus allowing linear approximation of the main rotor force components to be along the helicopter body axes.

13. The engine torque response to throttle changes can be considered instantaneous.

14. The engine governor is modeled as a proportional-integral feedback controller.

15. Throttle servo dynamics are much faster than the rotorspeed dynamics and are neglected.

2.3 Rotorcraft Equations of Motion

2.3.1 Rigid-body equations of motion

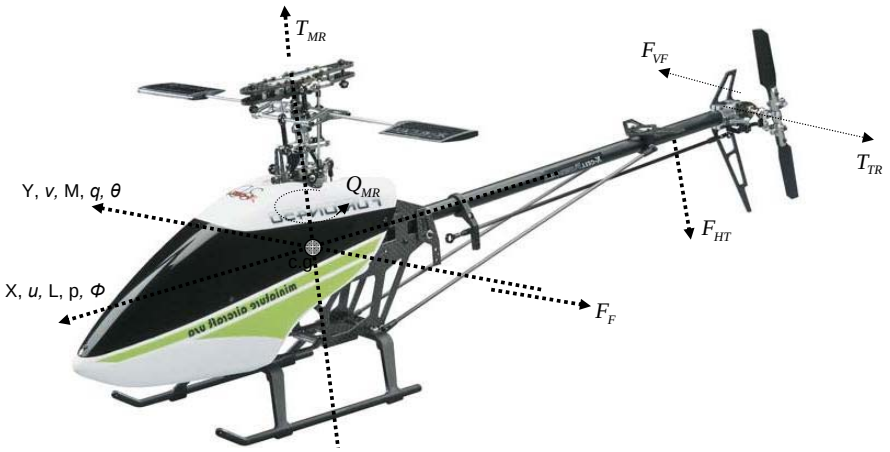

Figure 2.5: Moments and forces acting on the helicopter

The helicopter is a vehicle that is free to rotate and translate in all six degrees of freedom. Newton-Euler method is used to develop the rigid body equations. These equations are expressed in the inertial reference frame and derived from the principle of conservation of linear and angular momentum. For constant vehicle mass m and moment of inertia (inertia tensor **I**), they are:

$$m\frac{d^I v}{dt} = \mathbf{F} \tag{2.1}$$

$$\mathbf{I}\frac{d^I \omega}{dt} = M \tag{2.2}$$

where **F** = $[X \;\; Y \;\; Z]^T$ is the vector of external forces on the vehicle center of gravity and **M** = $[L \;\; M \;\; N]^T$ is the vector of external moments. Their orientations are shown in Figure 2.5. In a helicopter, the external forces and moments are produced by: the main rotor and the tail rotors representing the control forces and moments, the gravitational forces and the aerodynamic forces produced by the fuselage, vertical fin and horizontal stabilizer.

Using moving reference frame principles (Padfield, 1996), the equations of motion can be

expressed with respect to the body-fixed reference frame:

$$m\dot{v} + m(\omega \times v) = \mathbf{F} \tag{2.3}$$

$$\mathbf{I}\dot{\omega} + (\omega \times \mathbf{I}\omega) = \mathbf{M} \tag{2.4}$$

where $v = [u\ v\ w]^T$ and $\omega = [p\ q\ r]^T$ are the fuselage velocities and angular rates in the body-fixed frame, respectively. The angular orientation of the aircraft is described by Euler angles. Euler angles refer to a specific sequence of rotations about the vehicle body axes namely: the yaw angle ψ (about the z axis), pitch angle θ (about the 'new' y axis), and roll angle (ϕ about the new 'x' axis). The rotation matrix L_{BI} refer to the transformation of a vector coordinates in the inertial reference axis to vector coordinates in the helicopter body-fixed reference axis. This is given as:

$$L_{BI} = \begin{bmatrix} cos\theta cos\psi & cos\theta sin\psi & -sin\theta \\ sin\phi sin\theta cos\psi - cos\phi sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & sin\phi cos\theta \\ cos\phi sin\theta cos\psi + sin\phi sin\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi & cos\phi cos\theta \end{bmatrix} \tag{2.5}$$

The gravity vector in the inertial reference frame has the components, $\mathbf{g} = [0\ 0\ g]^T$. Its effect on the body reference axis can be written as:

$$\begin{aligned}
\dot{u} &= vr - qw - gsin\theta + (X_{mr} + X_{fus})/m \\
\dot{v} &= wp - ur + gsin\theta cos\phi + (Y_{mr} + Y_{fus} + Y_{tr} + Y_{vf})/m \\
\dot{w} &= uq - vp + gcos\phi cos\theta + (Z_{mr} + Z_{fus} + Y_{ht})/m \\
\dot{p} &= qr(I_{yy} - I_{zz})/I_{xx} + (L_{mr} + L_{vf} + L_{tr})/I_{xx} \\
\dot{q} &= pr(I_{zz} - I_{xx})/I_{yy} + (M_{mr} + M_{ht})/I_{yy} \\
\dot{r} &= pq(I_{xx} - I_{yy})/I_{zz} + (N_{vf} + N_{tr} - Q_e)/I_{zz} \\
\dot{\phi} &= p + (qsin\phi + rcos\phi)tan\theta \\
\dot{\theta} &= qcos\phi - rsin\phi \\
\dot{\psi} &= (qsin\phi + rcos\phi)sec\theta \\
\dot{x}_E &= u(cos\theta cos\psi) + v(sin\phi cos\psi - cos\phi sin\psi) + w(cos\phi sin\theta cos\psi + sin\theta sin\psi) \\
\dot{y}_E &= u(cos\theta cos\psi) + v(sin\phi cos\psi + cos\phi sin\psi) + w(cos\phi sin\theta cos\psi - sin\theta sin\psi) \\
\dot{z}_E &= usin\theta - v(sin\phi cos\theta) - w(cos\phi cos\theta)
\end{aligned}$$

The set of acting forces and moments are represented as follows: $()_{mr}$ for main motor, $()_{tr}$ for tail rotor, $()_{fus}$ for fuselage, $()_{vf}$ for the vertical fin and $()_{ht}$ for horizontal stabilizer. These are shown in Figure 2.5.

2.3.2 Rotor equations of motion

To account for rotor dynamics, we need simplified expressions for the rotor equations of motion. These equations will be used to express the rotor forces and moments acting on the helicopter center of gravity. In rotor aerodynamics, the blade of the rotor has a similar role as the wing of an airplane. The main difference occurs when the helicopter moves forward, the advancing blade sees a higher airspeed than the retreating blade causing asymmetry in the aerodynamics (Mettler, 2003).

The velocity at the blade is represented by a tangential and perpendicular component U_T and U_P respectively. The latter acts into the page of the hub plane. These velocity components contribute towards: rotor rotation about the hub, rotor inflow and the blade flapping motion.

The tangential velocity component at a blade station y can be written as:

$$U_T = \Omega y + (U_\infty cos\alpha_D) sin\Psi \tag{2.6}$$

The perpendicular velocity component is made of the rotor inflow velocity (described later) v_i, the free-stream velocity U_∞, the vehicle rotation (angular rates p and q) and the blade flapping rate:

$$U_P = U_\infty sin\alpha_D + v_i - y(psin\psi + qcos\psi) + y\dot{\beta} \tag{2.7}$$

The magnitude of the resultant velocity at the blade element is

$$U = \sqrt{U_P^2 + U_T^2} \tag{2.8}$$

The incremental lift produced by the blade element acting normal and in-line to the resultant airspeed respectively can be computed (Prouty, 1986)

$$dL = \frac{1}{2}\rho U^2 c C_{l\alpha} \alpha dy \tag{2.9}$$

$$dD = \frac{1}{2}\rho U^2 c C_{d\alpha} \alpha dy \tag{2.10}$$

c is the blade chord length and $C_{l\alpha}$ and $C_{d\alpha}$ are the airfoil lift and drag curve slopes respectively. The in-plane and out-of-plane forces components acting on the blade element can be deduced:

$$dF_z = dLcos\Phi + dDsin\Phi \approx dL \tag{2.11}$$

$$dF_x = dLsin\Phi + dDcos\Phi \approx dD \tag{2.12}$$

The total lift and drag forces are obtained by integrating the elemental forces along the blade length. The equations for the blade flapping motion are derived from the balance of moments about the flapping hinge. Figure 2.6 shows a rigid blade with the main forces and moments acting. The blade aerodynamic force F_{aero}, the centrifugal force F_{cent}, the inertial

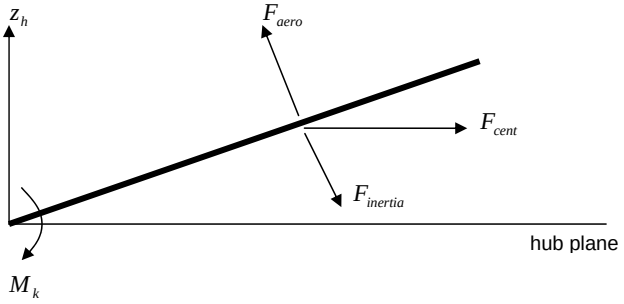

Figure 2.6: Forces and moments acting on the rotor blade.

forces $F_{inertia}$ (resulting from the rigid body equations) and the moment M_k produced by the flapping restraint.

The complete expression of the inertial and aerodynamic forces, not included here, typically consist of contributions from the vehicle angular and linear acceleration and the Coriolis acceleration. The elemental inertial force, produced the acceleration due to blade flapping, $F_{inertia} = mdy\ddot{\beta}$, the elemental centrifugal force component normal to the blade element is given as $F_{cent} = mdy\Omega^2 y\beta$. Writing the balance of moments acting on the flapping hinge, we obtain

$$\int_0^R ydF_z dy - \int_0^R y\left[my\ddot{\beta} - mdy\Omega^2 y\beta\right] dy - k_\beta\beta = 0 \tag{2.13}$$

Rearanging the terms

$$\int_0^R ydF_z dy - \int_0^R my^2 dy\left[\ddot{\beta} + \Omega^2\beta\right] - k_\beta\beta = 0 \tag{2.14}$$

By definition, the blade moment of inertia about the flapping hinge is given:

$$\int_0^R my^2 dy \overset{def}{\rightarrow} I_\beta \tag{2.15}$$

Differentiating Eq.2.14 with respect to the angular blade position Ψ and $\Psi = \Omega t$, we get

$$\ddot{\beta} = \Omega^2 \frac{\delta^2\beta}{\delta\Omega^2} = \Omega^2\beta'' \tag{2.16}$$

Substituting Eq.2.15 and 2.16 into Eq.2.14 gives:

$$I_\beta\Omega^2\left[\beta'' + \beta\right] + k_\beta\beta = \int_0^R ydF_z dy \tag{2.17}$$

Using the mass-spring-damper (MSD) system to rearrange the above equation as follows:

$$\beta'' + \left(1 + \frac{k_\beta}{I_\beta\Omega^2}\right)\beta = \frac{1}{I_\beta\Omega^2}\int_0^R ydF_z dy \tag{2.18}$$

where the coefficient of β is square of natural flapping frequency ratio λ_β (relative to the

rotor angular speed Ω):

$$\lambda_\beta^2 = 1 + \frac{k_\beta}{I_\beta \Omega^2} = \frac{\omega_\beta^2}{\Omega^2} \tag{2.19}$$

ω_β is the natural flapping frequency. It can be shown that for a teetering rotor ($k_\beta = 0$), the natural flapping frequency is equal to the rotor angular speed Ω ($\lambda_\beta = 1$).

2.4 Aerodynamic Model

The aerodynamic model is based on a linear state-space model developed from a combination of first-principle vehicle dynamics, physical insight and frequency-response analysis (Gavrilets et al., 2003).

2.4.1 Main rotor forces and moments

Thrust

Assuming steady and uniform inflow, the settling time of the inflow transients at hover is given:

$$\tau_\lambda = \frac{0.849}{4\lambda_{trim}\Omega_{mr}} \tag{2.20}$$

The thrust coefficient is given as:

$$C_T = \frac{T}{\rho(\Omega_{mr}R_{mr})^2\pi R_{mr}^2} \tag{2.21}$$

where T is the main rotor thrust. Then the following systems of equations can be solved iteratively:

$$\lambda_0 = \frac{C_T}{2\eta_w\sqrt{\mu_{mr}^2 + (\lambda_0 - \mu_{z_{mr}})^2}} \tag{2.22}$$

$$C_T^{ideal} = \frac{a_{mr}\sigma_{mr}}{2}\left(\theta_0\left(\frac{1}{3} + \frac{\mu^2}{2}\right) + \frac{\mu_{z_{mr}} - \lambda_0}{2}\right) \tag{2.23}$$

The thrust coefficient is limited by the maximum rotor thrust procuded by the engine:

$$C_T^{max} = \frac{T_{max}}{\rho(\Omega_{mr}R_{mr})^2\pi R_{mr}^2} \tag{2.24}$$

here

$$\mu_{mr} = \frac{\sqrt{(u - u_{wind})^2 + (v - v_{wind})^2}}{\Omega_{mr} R_{mr}} \tag{2.25}$$

$$\mu_{z_{mr}} = \frac{w - w_{wind}}{\Omega_{mr} R_{mr}} \tag{2.26}$$

$$\sigma_{mr} = \frac{2c_{mr}}{\pi R_{mr}} \tag{2.27}$$

It needs to be noted that the maximum thrust produced by the main rotor can be directly related to the weight of the helicopter as:

$$T_{max} = 2.5mg$$

Torque

The main rotor torque can be approximated as a sum of induced torque and torque due to the profile drag on the blades.

$$C_{Q_{mr}} = \frac{Q_{mr}}{\rho(\Omega_{mr} R_{mr})^2 \pi R_{mr}^3} = C_T(\lambda_0 - \mu_z) + \frac{C_{D_0}\sigma_{mr}}{8}\left(1 + \frac{7}{3}\mu_{mr}^2\right) \tag{2.28}$$

where $C_{Q_{mr}}$ is the main rotor torque coefficient, C_{D_0} is the profile drag coefficient of the main rotor blade. The profile drag variation with collective setting is small and ignored.

Main rotor moments and flapping dynamics

Based on the assumptions given is Section 2.2.1, the flapping motion is a 2π periodic function described as a Fourier series:

$$\beta(\Psi) = \beta_0 - \beta_{1c} cos\Psi - \beta_{1s} sin\Psi - \beta_{2c} cos2\Psi - \beta_{2s} sin2\Psi - ... \tag{2.29}$$

Ignoring the second and higher harmonics in the Fourier series:

$$\beta(\Psi) = \beta_0 - \beta_{1c} cos\Psi - \beta_{1s} sin\Psi \tag{2.30}$$

The stabilizer bar flapping contributes to the change of the main rotor blade pitch angle θ through a mechanical linkage:

$$\theta(\Psi) = \theta_0 + \theta_{lon} sin\Psi + \theta_{lat} cos\Psi + k_s\beta_s \tag{2.31}$$

The swashplate Bell mixer changes the cyclic pitch angle of both main rotor and the stabilizer bar. The stabilizer bar paddles are free to teeter about the rotor shaft. Hence, the stabilizer bar does not produce any significant force and moment on the rotor hub. The damping ratio of the flapping motion can be correlated to the Lock number of the blades considered (Gavrilets et al., 2003). The Lock number represents the ratio of aerodynamic to inertial forces defined as:

$$\gamma = \frac{\rho c a R^4}{I_\beta} \tag{2.32}$$

The main rotor blades have a much higher Lock number than the stabilizer bar blades producing a smaller time constant in their response to cyclic inputs. Based on the assumptions given in Section 2.2.1, lateral and longitudinal flapping dynamics by first-order equations can be deduced:

$$\dot{b_1} = -p - \frac{b_1}{\tau_e} - \frac{1}{\tau_e}\frac{\delta b_1}{\delta \mu_v}\frac{v - v_{wind}}{\Omega_{mr} R_{mr}} + \frac{B_{\delta_{lat}}}{\tau_e}\delta_{lat} \tag{2.33}$$

$$\dot{a_1} = -q - \frac{a_1}{\tau_e} - \frac{1}{\tau_e}\left(\frac{\delta a_1}{\delta \mu}\frac{u - u_{wind}}{\Omega_{mr} R_{mr}} + \frac{\delta a_1}{\delta \mu_z}\frac{w - w_{wind}}{\Omega_{mr} R_{mr}}\right) + \frac{A_{\delta_{lon}}}{\tau_e}\delta_{lon} \tag{2.34}$$

where $B_{\delta_{lat}}$ and $A_{\delta_{lon}}$ are effective steady-state lateral and longitudinal gains from the cyclic inputs to the main rotor flap angles. δ_{lat} and δ_{lon} are the lateral and longitudinal cyclic control inputs and τ_e is the effective rotor time constant for the rotor and the stabilizer bar.

The moments of inertia was determined by using the torsional pendulum tests. τ_e was approximated by estimating the hub torsional stiffness given as:

$$\tau_e = \frac{16}{\gamma_{fb}\Omega_{mr}} \tag{2.35}$$

where γ_{fb} is the stabilizer Lock number. Experiments showed that $B_{\delta_{lat}}$ and $A_{\delta_{lon}}$ grow with rotor speed given as:

$$B_{\delta_{lat}} = B^{nom}_{\delta_{lat}}\left(\frac{\Omega}{\Omega_{nom}}\right)^2 \tag{2.36}$$

$$A_{\delta_{lon}} = A^{nom}_{\delta_{lon}}\left(\frac{\Omega}{\Omega_{nom}}\right)^2 \tag{2.37}$$

where Ω_{nom} is the nominal operating rotor speed at 167 rad/s. The flapping due to translational velocity is caused by an increase of lift on the advancing blade with respect to the retreating blade thereby causing a moment with a 90 degree gyroscopic phase lag on the main rotor. The stabilizer bar dramatically reduces the flapping response to gusts (Gavrilets, 2003)

so a scaling coefficient had to be included. The longitudinal dihedral derivative is given as:

$$\frac{\delta a_1}{\delta \mu} = 2K_\mu \left(\frac{4\delta_{col}}{3} - \lambda_0 \right) \tag{2.38}$$

K_μ is the scaling coefficient to include the stabilizer effect. K_μ = 0.2 for this helicopter. The longitudinal and lateral dihedral derivatives are equal in magnitude and both cause the rotor to flap away from the incoming air.

$$\frac{\delta b_1}{\delta \mu_v} = -\frac{\delta a_1}{\delta \mu} \tag{2.39}$$

The upward heave movement of the helicopter causes a higher lift on the advancing blade which causes a moment on the rotor hub. The same stabilizer scaling coefficient is used:

$$\frac{\delta a_1}{\delta \mu_z} = K_\mu \frac{16\mu^2}{(1-\mu^2/2)(8\,|\mu| + a_{mr}\sigma_{mr})} \tag{2.40}$$

Rotor flapping is the dominant effect on rotor moments. The restraint is approximated using a linear torsional spring with constant stiffness coefficient K_β. This results in a longitudinal (pitch) and lateral (roll) moments:

$$M_{k,lon} = K_\beta a_1 \tag{2.41}$$

$$L_{k,lat} = K_\beta b_1 \tag{2.42}$$

Once flapping occurs, the rotor thrust vector tilts and contributes to the rotor-fuselage moments. Assuming the thrust vector tilts proportionally to the rotor flapping angles, the total main rotor pitch and roll moments can be deduced as:

$$M_{mr} = (K_\beta + Th_{mr})\, a_1 \tag{2.43}$$

$$L_{mr} = (K_\beta + Th_{mr})\, b_1 \tag{2.44}$$

where h_{mr} is the distance between the rotor head and the fuselage center of gravity.

Rotor forces

For small advance ratios (see Section 2.2.1), the rotor forces are given as:

$$X_{mr} = -T_{mr} a_1 \tag{2.45}$$

$$Y_{mr} = T_{mr} b_1 \tag{2.46}$$

$$Z_{mr} = -T_{mr} \tag{2.47}$$

2.4.2 Engine, governor and rotorspeed model

The rotorspeed dynamics can be modeled as:

$$\dot{\Omega} = \dot{r} + \frac{1}{I_{rot}} \left[Q_e - Q_{mr} - n_{tr} Q_{tr} \right] \tag{2.48}$$

where the main rotor torque is given as:

$$Q_{mr} = \frac{C_Q}{\rho \left(\Omega_{mr} R_{mr} \right)^2 \pi R_{mr}^3} \tag{2.49}$$

where Q_e is the engine torque (positive clockwise), Q_{tr} is the tail rotor torque, n_{tr} is the tail rotor gear ratio and I_{rot} is the total rotating inertia referenced to the main rotor defined as:

$$I_{rot} = 2I_{\beta_{mr}} + I_{es} n_{es}^2 + 2I_{\beta_{tr}} n_{tr}^2 \tag{2.50}$$

where $I_{\beta_{mr}}$ and $I_{\beta_{tr}}$ are the main and tail rotor blade inertias respectively. I_{es} is the inertia of the engine shaft and all components rotating at the engine speed, n_{es} is the engine gear ratio. Experiments have allowed I_{rot} to be estimated as (Gavrilets et al., 2003):

$$I_{rot} = 2.5 I_{\beta_{mr}} \tag{2.51}$$

2.4.3 Fuselage forces

Hover and forward flight speeds below the induced velocity at hover, the rotor downwash is deflected by forward and side velocity. This creates a force opposing the movement. The X and Y drag forces created by the fuselage can be expressed within this flight regime:

$$X_{fus} = S_x^{fus} \frac{1}{2} V_{imr}^2 \frac{u}{V_{imr}}$$

$$Y_{fus} = S_y^{fus} \frac{1}{2} V_{imr}^2 \frac{v}{V_{imr}}$$

where S_x^{fus} and S_y^{fus} are the effective drag areas for the fuselage in the X and Y directions respectively. However when the forward flight is higher than the rotor induced velocity, the fuselage drag is modelled as the drag on a flat plate (Gavrilets, 2003). The X and Y forces can thus be expressed as:

$$X_{fus} = S_x^{fus} \frac{1}{2} U_e^2 \frac{u}{U_e}$$

$$Y_{fus} = S_y^{fus} \frac{1}{2} U_e^2 \frac{v}{U_e}$$

where U_e is the trim speed. The above equations can be expanded into the following fuselage forces:

$$X_{fus} = -S_x^{fus} \frac{1}{2} \left(u - u_{wind}\right) V_\infty \tag{2.52}$$

$$Y_{fus} = -S_y^{fus} \frac{1}{2} \left(v - v_{wind}\right) V_\infty \tag{2.53}$$

$$Z_{fus} = -S_z^{fus} \frac{1}{2} \left(w - w_{wind} + V_{imr}\right) V_\infty \tag{2.54}$$

where $V_\infty = \sqrt{\left(u - u_{wind}\right)^2 + \left(v - v_{wind}\right)^2 + \left(w - w_{wind} + V_{imr}\right)^2}$ is the total velocity acting on the helicopter. The sign convention discussed above has been included. Based on the fuselage projection areas, the fuselage effective areas are given as:

$$S_y^{fus} = 2.2 S_x^{fus}$$

$$S_z^{fus} = 1.5 S_x^{fus}$$

2.4.4 Vertical fin forces and moments

The sideforce produced by the vertical fin is given as:

$$Y_{vf} = S_{vf} \left(C_{L\alpha}^{vf} V_\infty^{tr} + |v_{vf}| \right) v_{vf} \tag{2.55}$$

where S_{vf} is the vertical fin area and $C_{L\alpha}^{vf}$ is the fin lift curve slope. v_{vf} is the side velocity relative to the air at the fin location defined as:

$$v_{vf} = v - v_{wind} - \epsilon_{vf}^{tr} V_{itr} - l_{tr} r \tag{2.56}$$

the axial velocity V_∞^{tr} is defined as:

$$V_\infty^{tr} = \sqrt{(u - u_{wind})^2 + w_{tr}^2} \tag{2.57}$$

where

$$w_{tr} = w - w_{wind} + l_{tr} q - K_\lambda V_{imr} \tag{2.58}$$

V_{itr} is the induced velocity of the tail rotor, ϵ_{vf}^{tr} is the fraction of the vertical fin area exposed to full induced velocity from the tail rotor, l_{tr} is the vertical distance between the c.g. and

the tail velocity and K_λ is the wake intensity factor calculated in the tail rotor section Section 2.4.6. The vertical fin sideforce creates a yawing and rolling moment due to offsets from the c.g.

$$N_{vf} = -Y_{vf} l_{tr} \tag{2.59}$$

$$L_{vf} = Y_{vf} h_{tr} \tag{2.60}$$

where h_{tr} is the tail rotor hub distance from c.g.

2.4.5 Horizontal stabilizer forces and moments

The main rotor flapping produces a weathercock effect by the horizontal stabiliser. The horizontal tail produces a lift and stabilising pitching moment around the c.g. Assuming the stabiliser is fully or partially submerged in the downwash of the main rotor, an effective vertical speed on the horizontal tail can be determined as:

$$w_{ht} = w - w_{wind} + l_{ht} q - K_\lambda V_{imr} \tag{2.61}$$

where l_{ht} is the horizontal distance between the center of gravity of the horizontal tail and the helicopter. The same wake factor is used for the horizontal tail as for the vertical fin and tail rotor. The lift produced by the horizontal tail can be deduced:

$$Z_{ht} = \frac{1}{2} \rho S_{ht} \left(C_{L\alpha}^{ht} \left| u - u_{wind} \right| w_{ht} + \left| w_{ht} \right| w_{ht} \right) \tag{2.62}$$

where S_{ht} is the horizontal stabilizer area, $C_{L\alpha}^{ht} = 3.0$ is the lift curve slope of the horizontal stabiliser. To accommodate for horizontal stabiliser stall, the absolute value of the horizontal stabiliser lift is limited to:

$$|Z_{ht}| \leq \frac{1}{2} \rho S_{ht} \left((u - u_{wind})^2 + w_{ht}^2 \right) \tag{2.63}$$

The pitching moment can thus be determined as:

$$M_{ht} = Z_{ht} l_{ht} \tag{2.64}$$

2.4.6 Tail rotor

The tail rotor is subject to a wide range of flow conditions, including those where thrust inflow iteration algorithm fails (when the tail rotor operates within its own wake). Various aerodynamic coefficients are defined to evaluate the tail rotor thrust but the same approach for the main rotor computation in Section 2.4.1 is used here. In order to determine the normal μ_z^{tr} and the in-plane μ_{tr} tail rotor components, the effect of the main rotor wake on the tail

rotor needs to be determined. This is achieved through the following variables representing the geometric angles:

$$g_i = \frac{l_{tr} - R_{mr} - R_{tr}}{h_{tr}}$$

$$g_f = \frac{l_{tr} - R_{mr} + R_{tr}}{h_{tr}}$$

When the tail rotor is out of the main rotor downwash (when $V_{imr} \leq w - w_{wind}$), there is an effective upwash on the tail rotor blades. This is represented by the condition:

$$\frac{u - u_{wind}}{V_{imr} - (w - w_{wind})} \leq g_i \tag{2.65}$$

In this condition $K_\lambda = 0$. If the tail rotor is fully immersed in the main rotor wake, then the following condition needs to be satisfied:

$$\frac{u - u_{wind}}{V_{imr} - (w - w_{wind})} \geq g_f \tag{2.66}$$

In this condition $K_\lambda = 0$. If the tail rotor is partially immersed in the main rotor wake, the tail rotor upwash factor can be expressed as:

$$K_\lambda = \frac{\frac{u - u_{wind}}{V_{imr} - (w - w_{wind})} - g_i}{g_f - g_i} \tag{2.67}$$

A similar derivation as for the main rotor is used to calculate the tail rotor thrust coefficient:

$$C_T^{tr} = \frac{a_{tr}\sigma_{tr}}{2}\left(\delta_r\left(\frac{1}{3} + \frac{\mu_{tr}^2}{2}\right) + \frac{\mu_{tr} - \lambda_0^{tr}}{2}\right) \tag{2.68}$$

where

$$\mu_{tr} = \frac{\sqrt{(u - u_{wind})^2 + w_{tr}^2}}{\Omega_{tr} R_{tr}} \tag{2.69}$$

$$\mu_{z_{tr}} = \frac{v_{tr}}{\Omega_{tr} R_{tr}} \tag{2.70}$$

vertical component v_{tr} normal to the tail rotor is defined as:

$$v_{tr} = v - v_{wind} - l_{tr} r + h_{tr} p$$

the tail rotor thrust can be computed and its magnitude limited due to the blade stall as follows:

$$Y^{tr} = f_t C_T^{tr} \rho (\Omega_{tr} R_{tr})^2 \pi R_{tr}^2 \tag{2.71}$$

where f_t is the fin blockage factor (Padfield, 1996) defined as:

$$f_t = 1.0 - \frac{3S_{vf}}{4\pi R_{tr}^2} \tag{2.72}$$

It needs to be noted that $\Omega_{tr} = n_{tr}\Omega_{mr}$ where n_{tr} is the gear ratio given in Table.2.2. The yawing and rolling moment due to c.g. offsets can be computed:

$$N_{tr} = -Y_{tr}l_{tr} \tag{2.73}$$

$$L_{tr} = Y_{tr}h_{tr} \tag{2.74}$$

Similarly the tail rotor torque used in Eq. 2.48 can be defined as:

$$C_{Q_{tr}} = \frac{Q_{tr}}{\rho(\Omega_{tr}R_{tr})^2\pi R_{tr}^3} = C_T^{tr}(\lambda_0^{tr} - \mu_{z_{tr}}) + \frac{C_{D_0}^{tr}\sigma_{tr}}{8}\left(1 + \frac{7}{3}\mu_{tr}^2\right) \tag{2.75}$$

2.4.7 Actuation models

The actuation models were a mathematical representation of the servos used in the helicopter. Linear transfer functions were used to model the servo dynamics. The following transfer function has been developed (Gavrilets, 2003):

$$H_{servo}(s) = \frac{s/104 + 1}{s/33 + 1}\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{2.76}$$

The FUTABA S9402 servos were used for cyclic and collective input while the faster S9450 was used for the tail rotor pitch control. Time constants and damping ratio were determined through experiments. For the S9402, $\omega_n = 33\ rad/sec$ and $\zeta = 0.5$ while for the S9450, $\omega_n = 44\ rad/sec$ and $\zeta = 0.6$.

2.4.8 Sensor models

The PC computer boards can be used to construct the onboard computer stack installed on the underbelly of the helicopter shown in Figure.2.7. The four pieces of boards often includes: main processing board, A/D data acquisition board, serial communication board and DC-DC converter board.

Inertial measurement unit

The Inertial Measurement Unit (IMU) uses the gyros and accelerometers to measure angular rates, rotational and translational accelerations (resolution: $0.002g$ and 0.0027^o, data rate

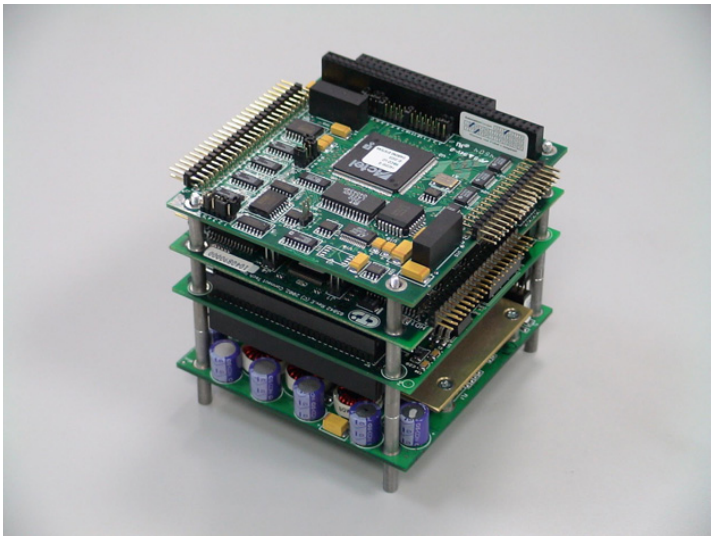

Figure 2.7: Onboard computer system

400Hz). The UM6 Orientation Sensor from CH Robotics is a typical example. A Global Positioning System (GPS) is often included to track the helicopter relative position and velocity (precision: 0.5^o, update rate 4Hz). Although the developed SIMULINK did not take into account rotational dynamics of the GPS box and feedback lags, it needs to be modelled when real flight data is collected.

2.4.9 Atmospheric model

The mathematical model of atmospheric turbulence are usually derived from power spectral functions (PSD). The power spectral density (PSD) of any function x(t) is a real function which provides information on how x(t) is mean squared value of x(t) is distributed with changing frequency. At any frequency ω, the PSD value represents the mean squared value of an infinitely small section of x(t) at that frequency. The PSD function is defined as:

$$\Phi(\omega) = \overset{lim}{\Delta\omega \to 0} T \to \infty \frac{1}{T\Delta\omega} \int_0^T x^2(t, \omega, \Delta\omega) \tag{2.77}$$

where $\Phi(\omega)$ - PSD function of and $x(t, \omega, \Delta\omega)$ is a component of x(t) which lies within specified frequency band.

Using the following relationship described as:

$$\Omega = \frac{\omega}{V_0} \tag{2.78}$$

where Ω and V_0 are the spatial frequency and aircraft speed respectively.

The original PSD function to one transformed in the new reference frame:

$$\Phi(\omega) = \frac{\Phi(\Omega)}{V_0} \tag{2.79}$$

There are two most common representations of PSD functions (McLean, 1990). The first is the Von Karman spectrum, which better represents the spectrum from records of atmospheric turbulence. But this method is very complex and not well suited for analytical purposes. The second is the Dryden PSD function which is more easily programmable and is used in our model.

$$\Phi_u(\Omega) = \frac{2\sigma_u^2 L_u}{\pi} \frac{1}{1 + (L_u\Omega)^2}$$

$$\Phi_v(\Omega) = \frac{2\sigma_v^2 L_v}{\pi} \frac{1 + 3(L_v\Omega)^2}{1 + (L_v\Omega)^2} \tag{2.80}$$

$$\Phi_w(\Omega) = \frac{2\sigma_w^2 L_w}{\pi} \frac{1 + 3(L_w\Omega)^2}{1 + (L_w\Omega)^2}$$

where:
$\sigma_u, \sigma_v, \sigma_w$ - standard deviations of u,v and w to define turbulence
L_u, L_v, L_w - scale lengths for power spectra

The above equations can also be expressed as:

$$\Phi_u(\omega) = \frac{2\sigma_u^2 L_u}{V_0\pi} \frac{1}{1 + (L_u/V_0\omega)^2}$$

$$\Phi_v(\omega) = \frac{2\sigma_v^2 L_v}{V_0\pi} \frac{1 + 3(L_v/V_0\omega)^2}{1 + (L_v/V_0\omega)^2} \tag{2.81}$$

$$\Phi_w(\omega) = \frac{2\sigma_w^2 L_w}{V_0\pi} \frac{1 + 3(L_w/V_0\omega)^2}{1 + (L_w/V_0\omega)^2}$$

The above PSD functions can be used to generate gust signals from specified scale lengths and intensity velocities. This is done constructing a filter which will take white noise as inputs and output an appropriate frequency response which will correspond to the given PSD function. This is illustrated in Figure 2.8 (McLean, 1990).

This relationship can be defined as:

$$\Omega_i(\omega) = |G_{j\omega}|^2 \, \Phi_N(\omega) \quad i = u, v, w \tag{2.82}$$

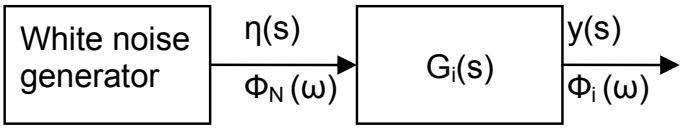

Figure 2.8: Gust signals from noise filter

The noise source is chosen to that similar to its power spectrum so that $\Phi_N(\omega) = 1$. Then:

$$\Omega_i(\omega) = |G_{j\omega}|^2 \quad i = u, v, w \tag{2.83}$$

This allows the filters to generate spectral densities for the translational gust velocities:

$$G_u(s) = \frac{\sqrt{K_u}}{s + \lambda_u} \tag{2.84}$$

$$G_v(s) = \frac{\sqrt{K_v}(s + \beta_v)}{(s + \lambda_v)^2} \tag{2.85}$$

$$G_w(s) = \frac{\sqrt{K_w}(s + \beta_w)}{(s + \lambda_w)^2} \tag{2.86}$$

where

$$\lambda_i = \frac{V_0}{L_i} \quad i = u, v, w$$

$$\beta_i = \frac{V_0}{\sqrt{3}L_i} \quad i = v, w$$

$$K_u = \frac{2V_0\sigma_u^2}{L_u\pi}$$

$$K_i = \frac{3V_0\sigma_i^2}{L_i\pi} \quad i = v, w$$

Taking Equation 2.84 as an example. Then,

$$w_g(s) = \sqrt{K_w}\frac{(s + \beta_w)}{(s + \lambda_w)^2}\eta(s) \tag{2.87}$$

where $\eta(t)$ is signal from the white noise source. From Equation 2.87 it can derived that:

$$\dot{u}_g + \lambda_u \dot{u}_g = \sqrt{K_u} \tag{2.88}$$

similarly,

$$\ddot{v}_g + 2\lambda_v \dot{v}_g + \lambda_v^2 v_g = \sqrt{K_v}\beta_v \eta + \sqrt{K_v}\dot{\eta} \tag{2.89}$$

$$\ddot{w}_g + 2\lambda_w \dot{w}_g + \lambda_w^2 w_g = \sqrt{K_w}\beta_w \eta + \sqrt{K_w}\dot{\eta} \tag{2.90}$$

McLean (1990) has shown that the above transfer functions can be expressed as state space equations to determine the turbulent velocities in the time domain. The wind shear effects can also be determined through approximating the rate of change of the translational velocities in pitch and yaw. The pitch rate is given as:

$$q_g = \frac{1}{V_0}\frac{dw_g}{dt} \tag{2.91}$$

and the yaw rate as:

$$r_g = \frac{1}{V_0}\frac{dv_g}{dt} \tag{2.92}$$

According to military references, the turbulence scale lengths at low altitudes ($\leq$ 303m), where h is the altitude in feet, is defined (Yeager, 1998):

$$L_w = h \tag{2.93}$$

$$L_u = L_v = \frac{h}{(0.177 + 0.000823h)^{1.2}} \tag{2.94}$$

The turbulence intensities σ, are given below where W_{20} is the wind speed at 20 feet (6m). For light turbulence, the wind speed is 15 knots (7.6 m/s), moderate turbulence the wind speed is 30 knots (15.3 m/s) and severe turbulence wind speed is 45 knots (22.9 m/s).

$$\sigma_w = 0.1W_{20} \tag{2.95}$$

$$\sigma_u = \sigma_v = \frac{\sigma_w}{(0.177 + 0.000823h)^{0.4}} \tag{2.96}$$

2.5 Simulation Model

A simulation facility for a miniature helicopter UAV model was developed within the MATLAB/SIMULINK software suite. Simulink uses a graphical user interface (GUI) whereby

mathematical model is created using block diagrams. The comprehensive block library enables the modelling and analysis of realistic nonlinear models thereafter using MATLAB programming for post-processing and visualization.

Representation of the rotorcraft equations of motion and the associated aerodynamic forces and moments was achieved using a modular approach. Figure 2.9 shows the different models of the unmanned helicopter namely: Servo actuator model; helicopter aerodynamic model; the yaw gyro model; the IMU model and GPS model. A detailed schematic of the aerodynamic model and equations of motion model is given in Figure 2.10 and Figure 2.11 respectively. The helicopter control effectors, used for trim analysis (Chapter 3) and model identification (Chapter 4), are described:

- δ_{lon} - longitudinal cyclic
- δ_{lat} - lateral cyclic
- δ_{col} - collective
- δ_t - engine thrust level
- δ_r - yaw pedal

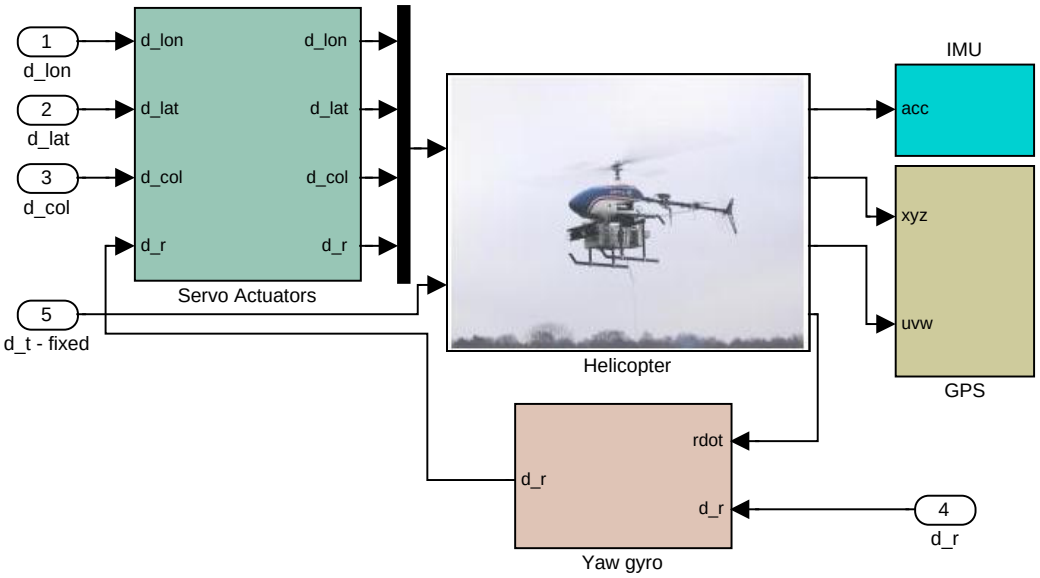

Figure 2.9: X-Cell.60 SIMULINK simulation model.

The helicopter aerodynamic model is comprised of the following subsystem models: equation of motion model; main rotor model; tail rotor model; horizontal and vertical stabiliser model; fuselage model; engine and governor model and atmosphere model. The aerodynamic forces (X, Y, Z) and moments (L, M, N) from each model then summed together and

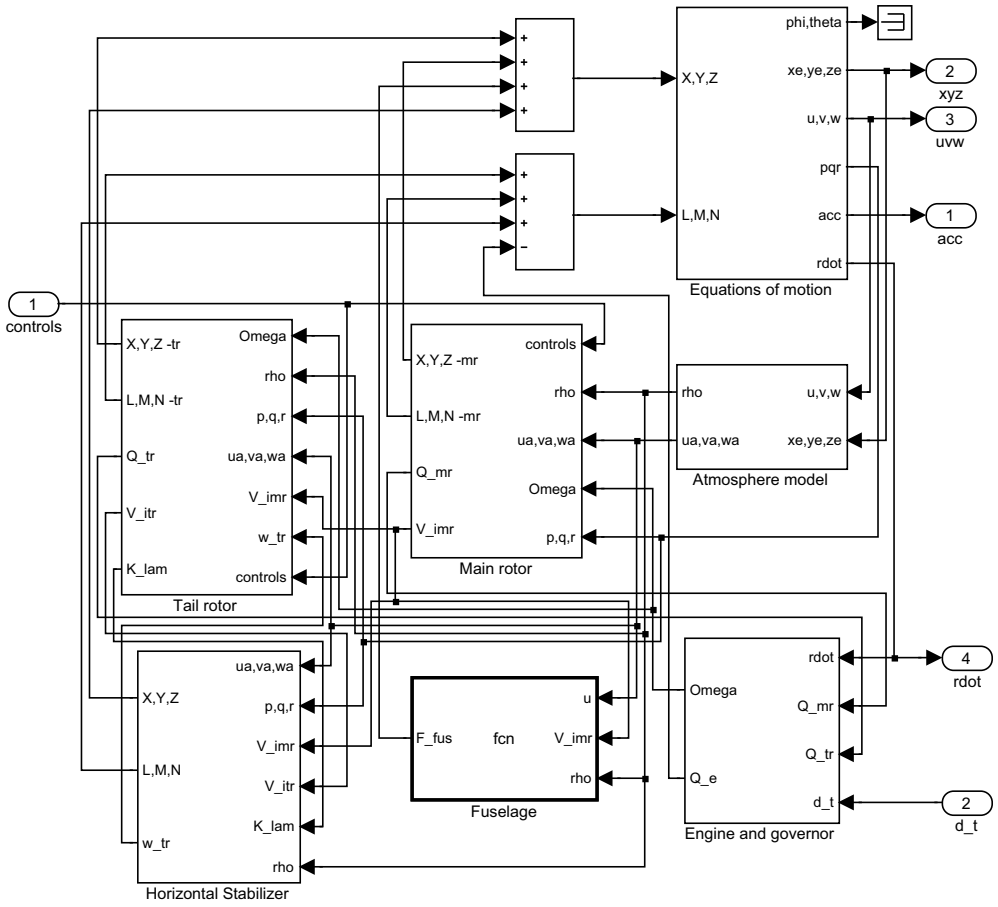

Figure 2.10: Helicopter aerodynamic model subsystem.

used in the equation of motion model. The yaw gyro model made use of a PID controller with the yaw rate r and yaw rotational acceleration $\dot{r}$ as state inputs.

Validation of the simulation model was achieved during the trim and stability analysis whereby time simulations of the state variables were monitored and compared with literature (Gavrilets et al., 2003; Mettler, 2003). In hover flight, a direct comparison of the generated main rotor thrust and the helicopter weight was used as a validation criterion.

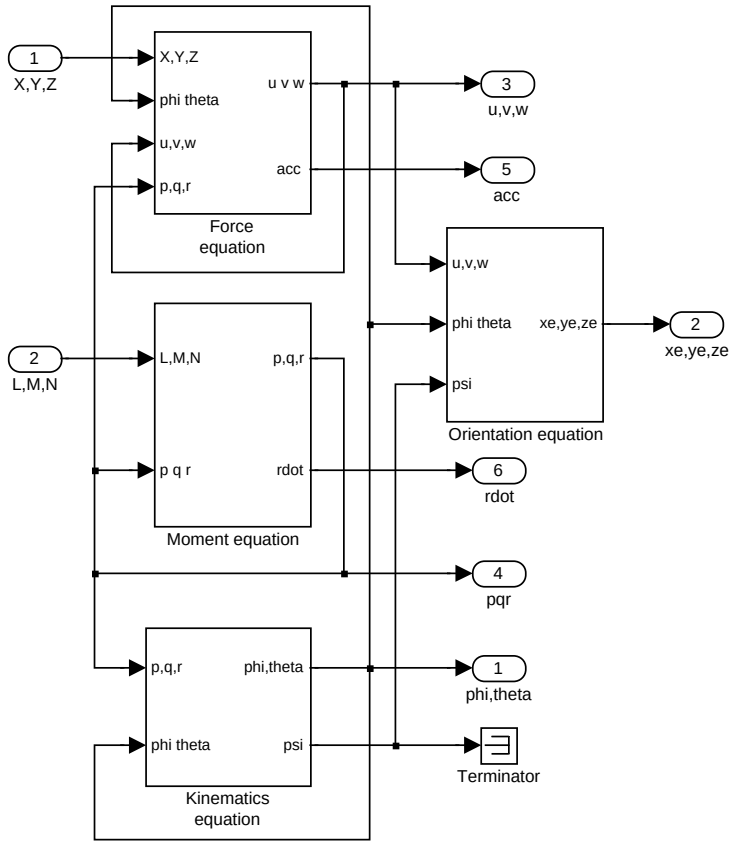

Figure 2.11: Helicopter equations of motion subsystem.

Chapter 3

Trim and Stability Analysis

3.1 Trim Analysis

Steady flight occurs when the helicopter is in equilibrium with respect to three forces and three moments acting along/around the three orthogonal axes through its center of gravity. This is called a *trim point* (Prouty, 1986). A trim point exists in the parameter space of a dynamic system at which the system is in a steady state. So in the case of a helicopter, the trim point in hover or forward flight will require the collective, cyclic and throttle setting to produce enough thrust to counteract the weight components and the inflow drag forces produced on the fuselage.

3.1.1 Hover flight

Helicopter hovering is the most challenging part of flying a helicopter. In practical term, it is impossible to keep the helicopter on fixed point in space as it requires constant control inputs. This unstable behaviour is primarily caused by helicopter rotor wake which by principle is a dynamic effect. This is shown in Figure 3.1.

Assuming a linear and constant rotor wake, the following forces need continuous adjustment for the helicopter to remain in a hovering flight condition:

- The collective/throttle setting is adjusted such that the vertical component of the main rotor thrust counteracts the weight of the helicopter and the downwash force produced by the drag on the fuselage.
- The tail rotor thrust produces a moment to counteract and it is directly linked to the main rotor via gears/belt.
- The tail rotor side force is counteracted by the sideways tilting of the main rotor using the cyclic control. This is done to prevent sideways drift.

Figure 3.1: Helicopter rotor wake.

- The main rotor tilting induces a moment on the fuselage causing the upthrust from the tail rotor.
- This upthrust is counteracted by the backward tilting of the main rotor using the cyclic control.

It can be deduced that any disturbances to this delicate balancing will require constant inputs from the pilot.

3.1.2 Forward flight

In forward flight, the rotor disk velocity distribution is asymmetric. The advancing side of the rotor disc sees a combination of rotor speed and forward airspeed (movement through the air mass) which is faster than the retreating side, which sees a combination of rotor speed and a 'reduced' forward airspeed (Prouty, 1986). This produces an uneven lift distribution creating a rolling moment. This is illustrated in Figure 3.2.

The ingenious method of equalizing this asymmetry of lift in forward flight is to allow the blades to flap. By connecting the blades to the hub by a method which allows a flexible up-down motion, the advancing blade, which encounters higher lift, begins to flap upward. The retreating blade, which encounters less lift, flaps downward (Prouty, 1986).

The increase of the forward speed introduces reverse flow on the retreating blade. The compressibility effects on the advancing blade and airfoil stall on the retreating blade, define the maximum forward speed of the helicopter (Kaletka and Hamel, 1997).

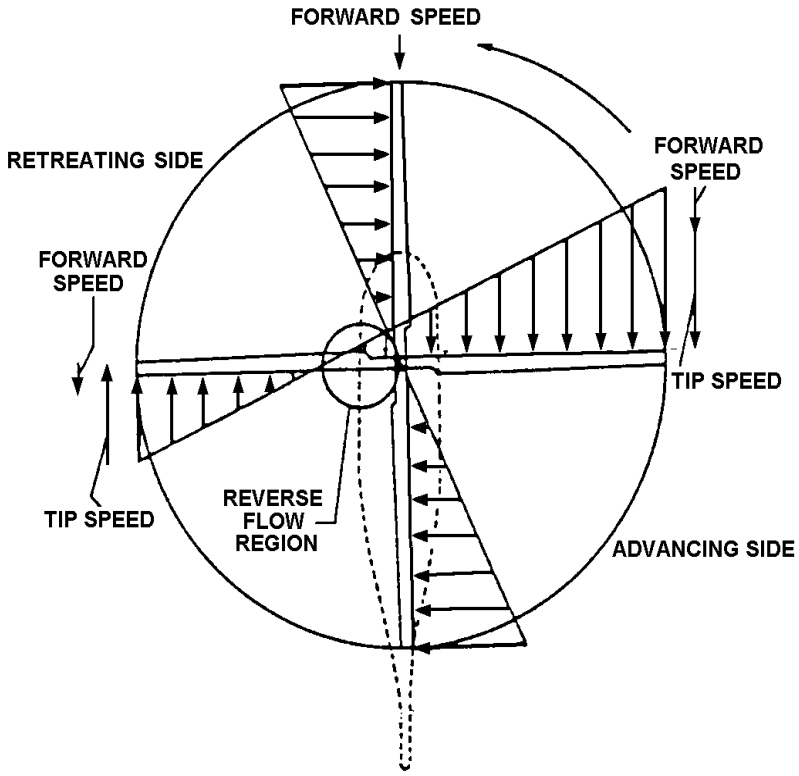

Figure 3.2: Rotor Aerodynamics in forward flight.

3.1.3 Trim procedure

The helicopter model was trimmed using the SIMULINK Control and Estimation Tools Manager. The following trim points were identified and defined:

- Hover Flight - all state variables are set to zero
- Forward Flight 10m/s - all state variables are set to zero except the body forward velocity $u = 10m/s$
- Forward Flight 20m/s - all state variables are set to zero except the body forward velocity $u = 20m/s$

Trim specifications are set in table form for each trim points. The known values of the system states, inputs and outputs are set and the desired steady state signals chosen. Furthermore, the maximum and minimum values of the system states, inputs and outputs are also specified. The gradient descent algorithm was the optimization method used for trim computation. This enforces a constraint on the output signals and force time derivatives of states to zero.

3.2 Stability Analysis

3.2.1 Model linearization

The helicopter model was linearized using the SIMULINK Control and Estimation Tools Manager. Linearization refers to the function or system linear approximation at a trim point. This is a method of accessing the local stability around an equilibrium point of a nonlinear differential equations system. Therefore, linearization also simplifies the analysis of nonlinear systems by studying its equivalent linear system near a given point. A system defined as:

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}, t) \tag{3.1}$$

the linearized system can be written as

$$\frac{d\mathbf{x}}{dt} = D\mathbf{F}(\mathbf{x}_0, t) \cdot (\mathbf{x} - \mathbf{x}_0) \tag{3.2}$$

where $\mathbf{x}_0$ is the point of interest (usually trim point) and $D\mathbf{F}(\mathbf{x}_0)$ is the Jacobian of $\mathbf{F}(\mathbf{x})$ evaluated at $\mathbf{x}_0$

The helicopter model was linearized for hover, 10m/s and 20m/s forward flight. Selection of input and output linearization points (also called analysis Input-Output's) was done to configure the model prior to linearization. Using numerical perturbation, linearization is achieved through the tracking of a signal traveling between an input and output point. This is based on the operating or trim points defined during the trim analysis in Section 3.1.

The *'block by block analytic'* linearization algorithm was used. It linearizes the model blocks individually and then combines the results for the whole system linearization. This method allows the designer to customize the linearization problem by user-defining the system inputs and output markers as shown in Figure.3.3.

The linearized model has the following control inputs:

$$\delta_{lon}, \delta_{lat}, \delta_{col}, \delta_r, \delta_t$$

the states:

$$u, v, w, \phi, \theta, \psi, p, q, r, x_e, y_e, z_e, a_1, b_1, \Omega, \omega_i, r_i$$

where a_1, b_1, Ω, ω_i and r_i are the longitudinal and lateral flapping angles, the rotorspeed, the engine controller and yaw gyro integrator state values respectively. This has been reduced to measured states from the flight instrumentation system:

$$u, v, w, \phi, \theta, \psi, p, q, r, a_1, b_1$$

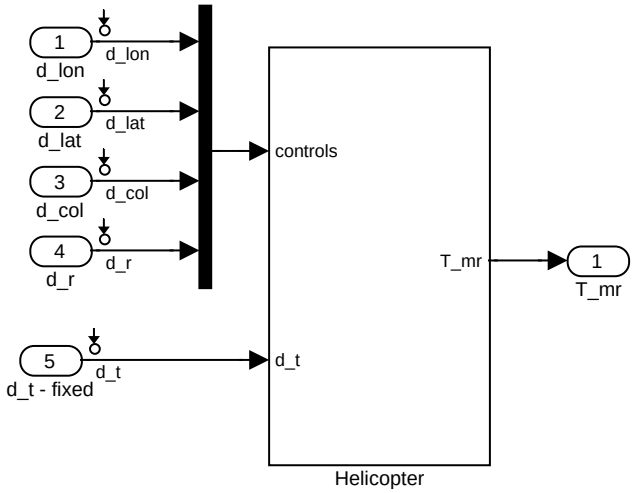

Figure 3.3: Helicopter linearization model

where a_1 and b_1 the longitudinal and lateral flapping angles respectively. The system outputs are:

$$\dot{u}, \dot{v}, \dot{w}, u, v, w, \dot{p}, \dot{q}, \dot{r}, p, q, r$$

The outputs were based on the measured variables from an Inertial Measurement Unit (IMU) and GPS sensor. The linear system is defined in terms of the following state space equation (excluding disturbances):

$$\dot{x} = \mathbf{A}x + \mathbf{B} \tag{3.3}$$

$$y = \mathbf{C}x + \mathbf{D} \tag{3.4}$$

where **A,B,C** and **D** are the system matrices.

3.2.2 Linearization procedure

Using *'block by block analytic'* algorithm, the model diagram was configured by selecting the input and output linearization points. Each trim point is chosen about which to linearize the system.

The fully parametrized linear system is represented using state space matrix form. Listing of the state space matrices for hover, forward 10 and 20 m/s flight is given in Appendix A on page 180. The system linearization has two purposes:

- Establishing the sign notation of the computed parameters. This is used as a model validation criterion (see Chapter 2).
- Defining a parameter database as a baseline during the system identification process (see Chapter 4).

3.2.3 Natural modes of motion

A natural mode is a pattern of motion in which most parts of the system move sinusiodally with the same frequency and damping factor. These frequencies of the normal modes are known as its natural frequencies. The modes are orthogonal to each other, that is based on the assumption that an excitation of one mode will never cause motion of a different mode. These excitations leads to large amplitudes of displacements which are often destructive (Prouty, 1986).

Given a system matrix, the eigenvalues λ are computed in such a way:

$$\mathbf{Ax} = \lambda \mathbf{x} \tag{3.5}$$

where A is an n-by-n matrix, x is a length n column of eigenvector and λ is a scalar length n vector of eigenvalues.

The eigenvalues which represents the characteristic behaviour of the model, are given as:

$$\lambda = n \pm i\omega \tag{3.6}$$

where ω_n is the undamped natural frequency is given as:

$$\omega_n = \sqrt{n^2 + \omega^2} \tag{3.7}$$

the damping ratio can also be computed:

$$\zeta = \frac{-n}{\omega_n} \tag{3.8}$$

The period of frequency can be calculated.

$$T = \frac{2\pi}{\omega_n} \tag{3.9}$$

The time to double or half the amplitude of oscillation is defined for conjugate pair or real:

$$T_{1/2} = \frac{ln2}{abs(n)} \tag{3.10}$$

3.3 Results

The trim points were found by simultaneously solving the six degrees of freedom equations of motion, the main rotor force and moments equations, the tail rotor side force equation and the two quasi-steady blade flapping equations. Model linearization was then achieved about each trim points and the state space matrices defined.

The simulation model was trimmed for hover, 10 m/s and 20 m/s flight at an altitude of 80m above ground. Calm wind conditions were considered with negligible atmospheric disturbances. Table 3.1 - 3.3 show the trim values for the various flight conditions. The rotor speed of 167 m/s was achieved for all flight conditions. The rotor disc plane has a rolling angle of approx. 0.74 deg (0.0132 rad) starboard side in the hover flight condition. This is in agreement with conventional helicopter dynamics as the main rotor thrust component is opposite and equal to the tail rotor thrust in order to prevent sideways movement. As the forward flight speed increases from 0 to 20m/s, the body pitch angle decreases from 0 to approx -19.4 deg (-0.3378 rad) and the rolling angle increases from 0.74 deg (0.0132 rad) to 1.04 deg (0.0181 rad). The flapping dynamics were also confirmed in forward flight as the longitudinal flapping angle increased to a value of 1.56 deg (0.0273 rad). This can be attributed to the asymmetric lift distribution producing a 90 degree-lag force on the main rotor blade.

These trim values were used in the nonlinear simulation model for validation. Hover flight trim (Figure 3.4); Forward flight 10m/s (Figure 3.5); Forward flight 20m/s (Figure 3.6). The translational velocities (u, v, w), attitude Euler angles (ϕ, θ, ψ), rotor flapping angles (a_1, b_1), rotational rates (p, q, r) and Inertial reference c.g. position (x_E, y_E, z_E) are simulated. The hover flight results showed that a helicopter in hover trim is a quasi-steady condition as the translational velocities are not in steady state shown in Figure 3.4. The yaw rate ψ peaking at 0.04 deg/s which could be the result of the yaw rate gyro proportional 'over correcting' through tail rotor thrust. The 10m/s and 20 m/s forward flight body rates in Figures 3.5 and 3.6 respectively, show increasing divergence specifically in the yaw rate. This is mainly due to numerical errors as the helicopter behaves like an aircraft in straight and level flight. This is confirmed by the helicopter inertial reference positions for 10m/s and 20m/s forward flight.

The natural modes (eigenvalues) for the linearized system were calculated for the hover flight, 10m/s and 20m/s forward flight. These are shown in Tables 3.4, 3.5 and 3.6 respectively. From these results, the natural frequency, damping ratio, period of oscillation and time to double or half were calculated. The hover flight condition shows four unstable modes namely mode 7 and mode 8 and mode 9-10. Mode 7, the strongest unstable mode, is excited in the lateral-directional plane with the yaw rate r and yaw angle ψ showing strong inter-dependencies also seen in mode 8. Mode 9-10 represents an unstable Phugoid mode characterized by light damping (-0.102) and low natural frequency (0.157). The large time to double (43.2 sec) is indicative of the minimal pilot effort required to escape this unfavorable condition. This is confirmed as helicopter body pitch θ and the forward velocity u lag each other by 90^o.

The high frequency (approx 19 rad/s) and lightly damped (approx 0.28) modes 2-3 and modes 4-5 corresponds to the coupled fuselage/flapping/stabilizer-bar modes in the lateral and longitudinal planes respectively. The small damping directly reflects the large rotor time constant. This stays fairly constant from hover to high-speed forward flight consistent with the swashplate/stabiliser bar physical properties. The strongly-coupled fuselage/flapping

modes emphasize the importance of the rotor dynamics. The heave mode (mode 6), appears in hover flight but is replaced by an unstable mode with value of 1.416. This confirms the transformation of the helicopter dynamics from hover to forward flight. The absence of the phugoid mode in Table 3.5 is due to the tail rotor being partially immersed in the main rotor wake producing strong damped response to mitigate any directional oscillations. This is confirmed by mode 6 to mode 8. The 20m/s forward flight and the hover modes are similar in nature except the phugoid mode is stable which can be associated to fixed-wing dynamics. The yaw dynamics, present throughout the flight conditions can also be seen in mode 6 to mode 8. Mode 9-10 contains strong coupling between the translational velocities (u, v, w). This represents the yaw-roll coupling inherent in fixed-wing dynamics.

Table 3.1: Hover trim parameter values.

Parameter	Value	Parameter	Value
ω_{mr}	167 rad/s	p	0 rad/s
u	0 m/s	q	0 rad/s
v	0 m/s	r	0 rad/s
w	0 m/s	x_E	0 m
ϕ	0.0132 rad	y_E	0 m
θ	0 rad	z_E	50 m
ψ	0 rad	a_1	0 rad
		b_1	0.0013 rad

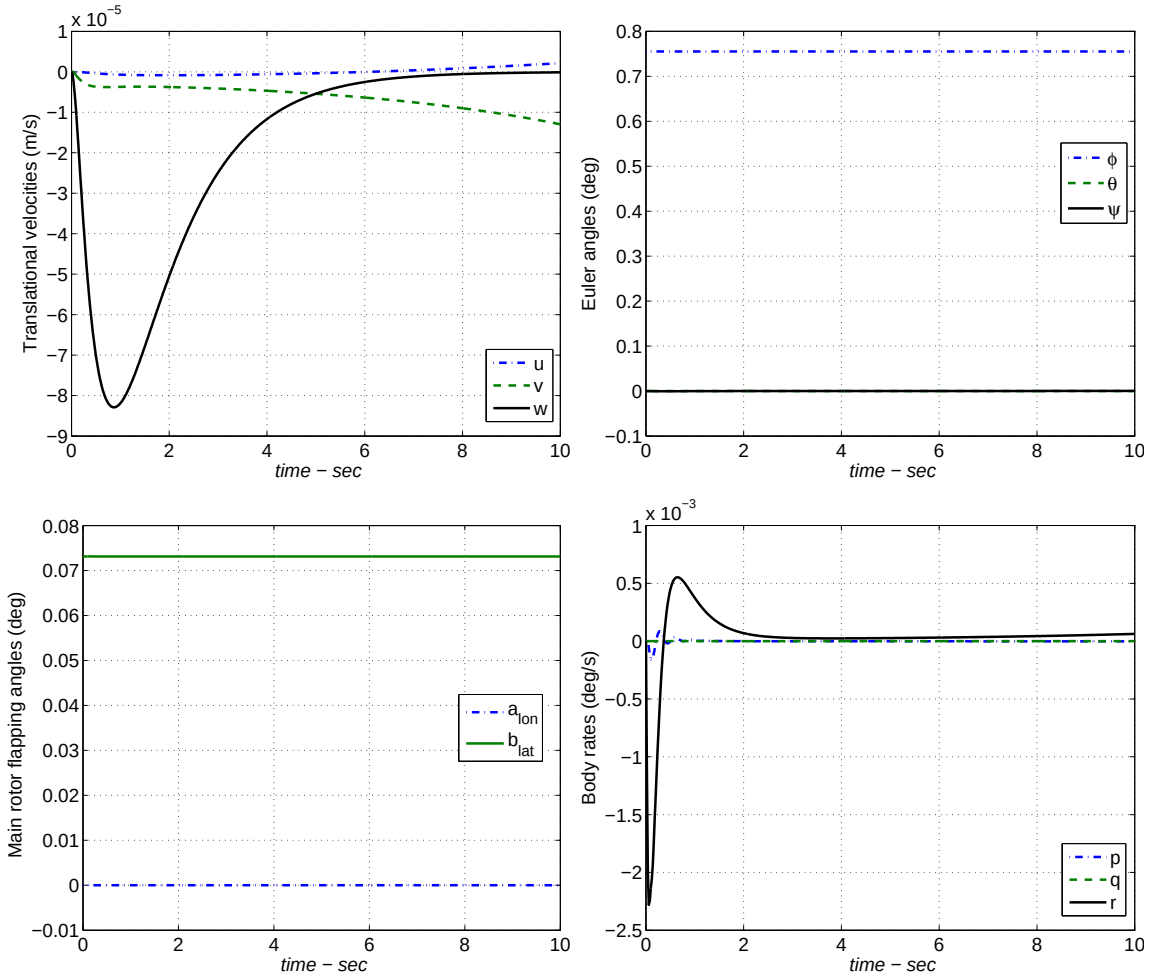

Figure 3.4: Hover trim: (1) body velocities. (2) body Euler angles. (3) Main rotor flapping angles. (4) body rates.

Table 3.2: Forward flight 10m/s trim parameter values.

Parameter	Value	Parameter	Value
ω_{mr}	167 rad/s	p	0 rad/s
u	10 m/s	q	0 rad/s
v	-0.0013 m/s	r	0 rad/s
w	-0.8168 m/s	x_E	0 m
ϕ	0.0118 rad	y_E	0 m
θ	-0.0815 rad	z_E	50 m
ψ	-0.0008 rad	a_1	0.0015 rad
		b_1	0.0011 rad

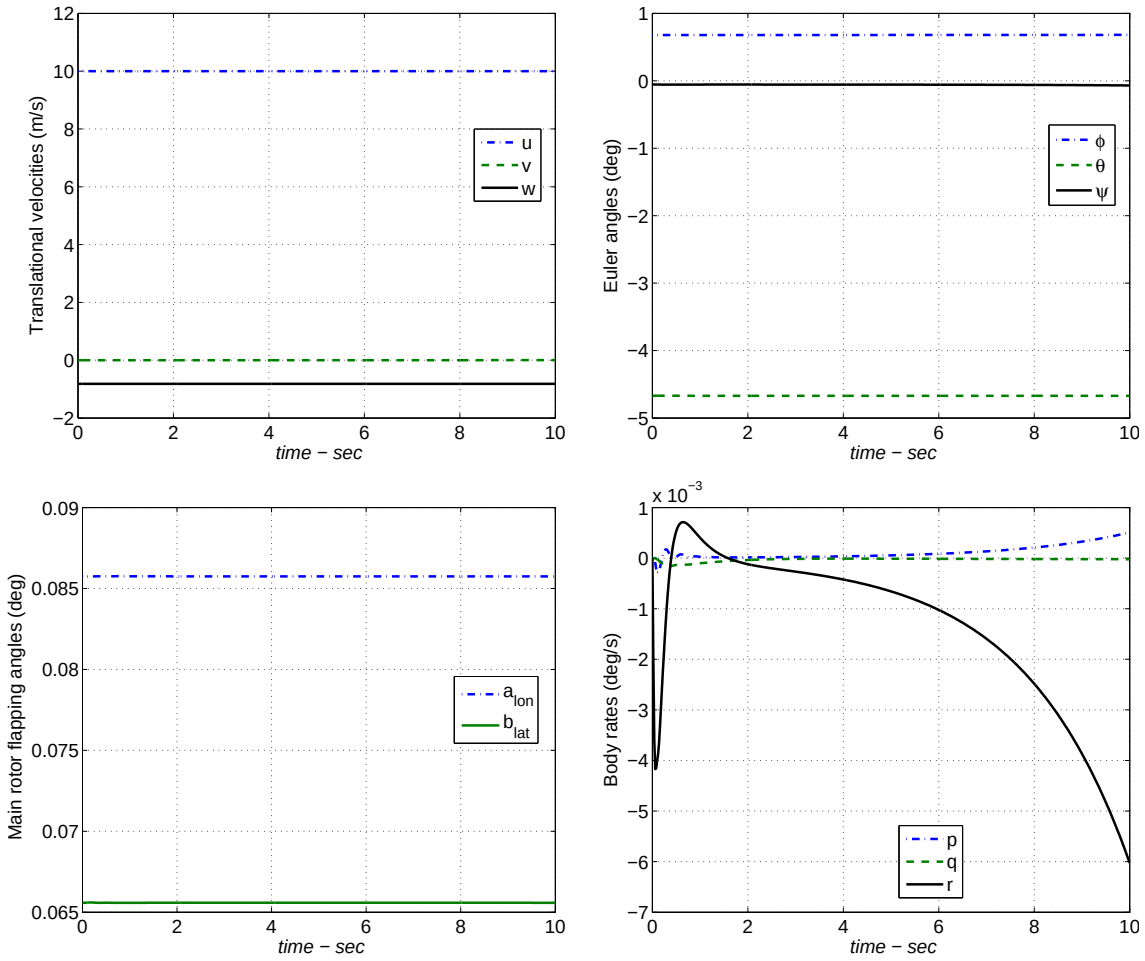

Figure 3.5: Forward flight 10m/s trim: (1) body velocities, (2) body Euler angles, (3) Main rotor flapping angles, (4) body rates.

Table 3.3: Forward flight 20m/s trim parameter values.

Parameter	Value	Parameter	Value
ω_{mr}	167 rad/s	p	0 rad/s
u	20 m/s	q	0 rad/s
v	-0.1264 m/s	r	0 rad/s
w	-7.0245 m/s	x_E	0 m
ϕ	0.0181 rad	y_E	0 m
θ	-0.3378 rad	z_E	50 m
ψ	0 rad	a_1	0.0273 rad
		b_1	0.0016 rad

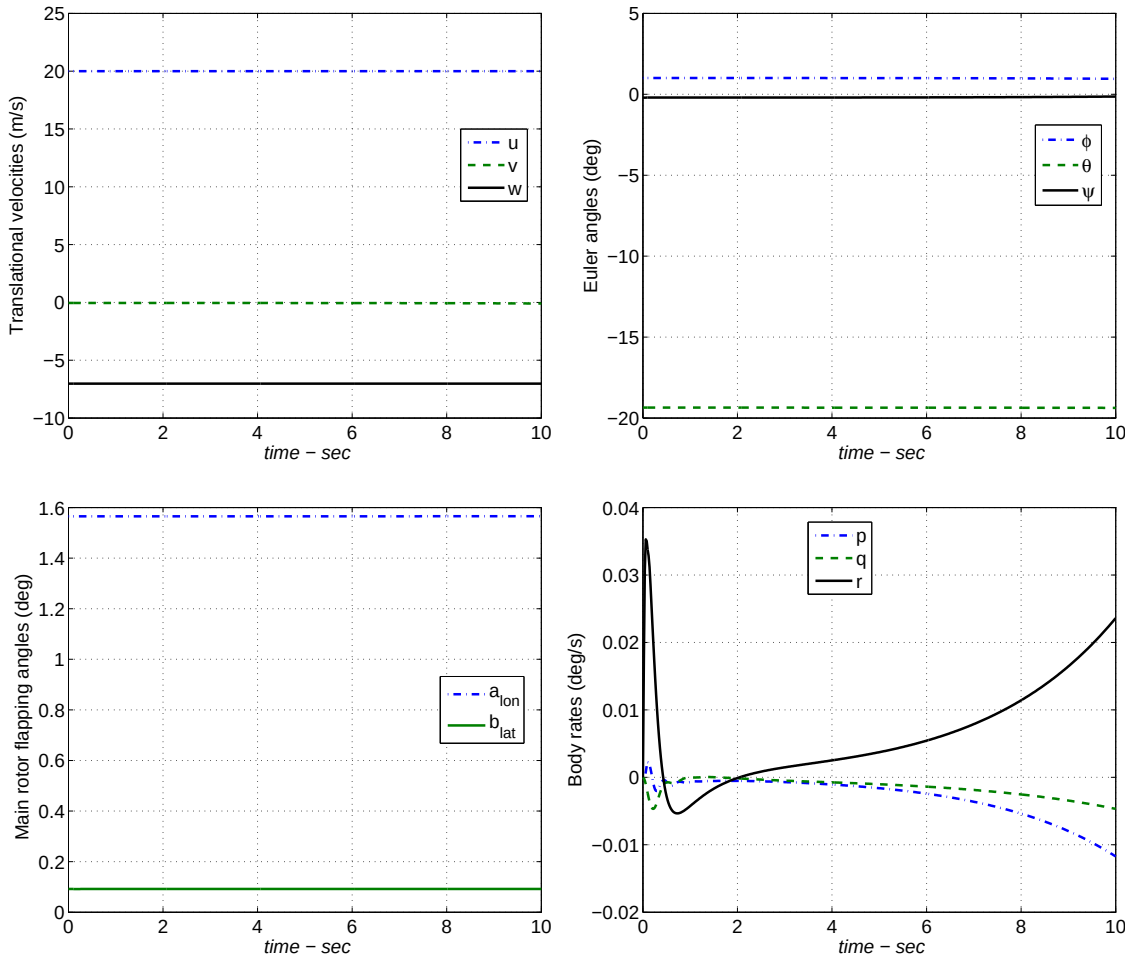

Figure 3.6: Forward flight 20m/s trim: (1) body velocities, (2) body Euler angles, (3) Main rotor flapping angles, (4) body rates.

Table 3.4: Hover flight natural modes

Flight mode	Eigenvalues	Damping ratio	Natural Frequency	Period (s)	Time to half/double (s)
Mode 2-3	-4.172 $\pm$ 19.687i	0.207	20.124	0.312	0.166
Mode 4-5	-4.175 $\pm$ 14.035i	0.285	14.643	0.429	0.166
Mode 6	-0.775	-	-	-	0.894
Mode 7	0.476	-	-	-	1.455
Mode 8	0.161	-	-	-	4.299
Mode 9-10	0.016023 $\pm$ 0.15651i	-0.102	0.157	39.937	43.261
Mode 11	-0.135	-	-	-	5.144

Table 3.5: Forward flight 10m/s natural modes

Flight mode	Eigenvalues	Damping ratio	Natural Frequency	Period (s)	Time to half/double (s)
Mode 2-3	-4.171 $\pm$ 19.673i	0.207	20.110	0.312	0.166
Mode 4-5	-4.104 $\pm$ 13.847i	0.284	14.442	0.435	0.169
Mode 6	1.416	-	-	-	0.489
Mode 7	-1.121	-	-	-	0.618
Mode 8	-1.405	-	-	-	0.493
Mode 9	-0.247	-	-	-	2.803
Mode 10	-0.078	-	-	-	8.845
Mode 11	0.085	-	-	-	8.181

Table 3.6: Forward flight 20m/s natural modes

Flight mode	Eigenvalues	Damping ratio	Natural Frequency	Period (s)	Time to half/double (s)
Mode 2-3	-4.155 $\pm$ 19.441i	0.209	19.880	0.316	0.167
Mode 4-5	-4.227 $\pm$ 13.633i	0.296	14.274	0.440	0.164
Mode 6	1.416	-	-	-	0.639
Mode 7	0.296	-	-	-	2.341
Mode 8	-0.161	-	-	-	4.295
Mode 9-10	-0.808 $\pm$ 0.112i	0.991	0.816	7.700	0.857
Mode 11	-1.514	-	-	-	0.458

Chapter 4

Online Model Identification and Parameter Estimation

4.1 Flight Vehicle System Identification

Flight vehicle system identification is a highly versatile procedure which is used to rapidly and efficiently extract dynamic models that best characterizes the measured response to controls. Within the online or recursive parameter estimation, the system identification problem is to estimate the parameter values for a pre-defined model structure (Kaletka and Hamel, 1997).

System identification plays an important role in flight vehicle development, whereby accurate and validated mathematical models are required within the following (Jategaonkar, 2006):

- Investigation of system performance and characteristics.
- Verification of wind-tunnel and analytical predictions.
- Development of high-fidelity aerodynamic databases for flight simulators meeting Airworthiness Authority fidelity requirements.
- Support of flight envelope expansion during prototype testing.
- Derivation of high-fidelity and high-bandwidth models for in-flight simulators.
- Design of flight control laws which include stability augmentation systems.
- Reconstruction of flight path trajectory, including wind estimation and incidence analysis.
- Fault-diagnosis and adaptive control or reconfiguration.
- Analysis of handling qualities specification compliance.

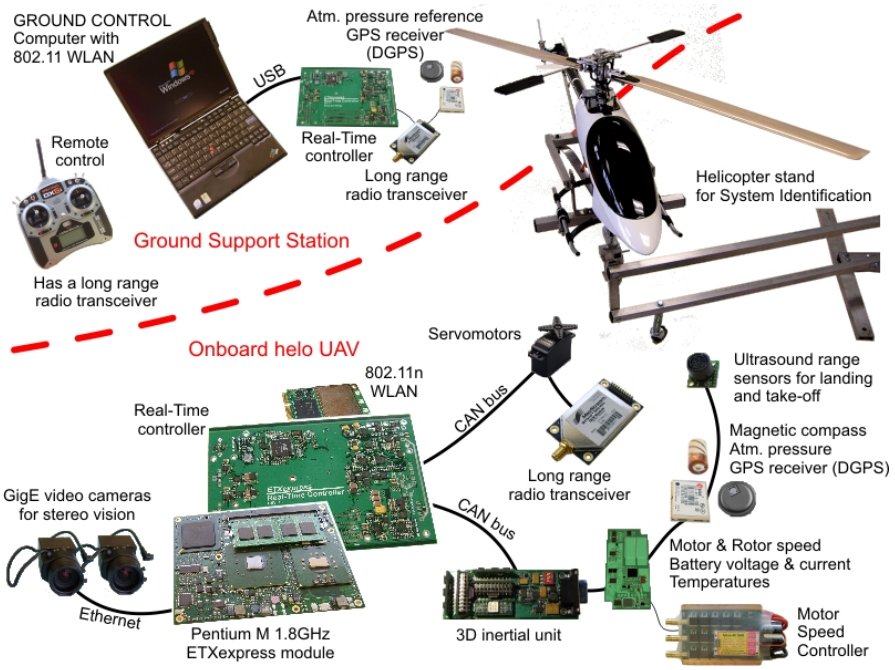

Figure 4.1: Instrumentation for Helicopter UAV System Identification

Different challenges have been found in rotorcraft system identification as the measured data exhibits reduced signal-noise ratio especially for low-speed and hovering flight regimes. These noise contributions arise from vibrations such as: rotor, engine, drive train and unsteady inflow mass through the rotor (Tischler and Remple, 2006). Specialized flight-test maneuvers have been developed to excite particular dynamics for applications within: flight dynamics and control, state and parameter estimation. The general approach to aircraft system identification requires both the inputs and aircraft response to be measured and recorded. In rotorcraft system identification, optimized control inputs and highly reliable measurements are indispensable towards achieving unbiased estimation results (Kaletka and Hamel, 1997).

4.1.1 Flight test instrumentation

The instrumentation of any flight test vehicle represents a significant investment of resources. Considerable care must be applied to identify the various instruments to ensure that the flight test instrumentation system will yield the required information. As shown in Figure 4.1 (Stingu and Lewis, 2008), a typical flight test instrumentation list consists of: radio transceivers, IMU, compass, GPS receiver, real-time controllers and a ground control station.

4.1.2 Flight test techniques

Apart from the reliability of the test instruments, system identification process relies on the information content of the system under test, provided by the measured control inputs and the resulting measured system responses. The test input, executed by various flight test techniques is one of the major factors influencing the accuracy of the model parameters to

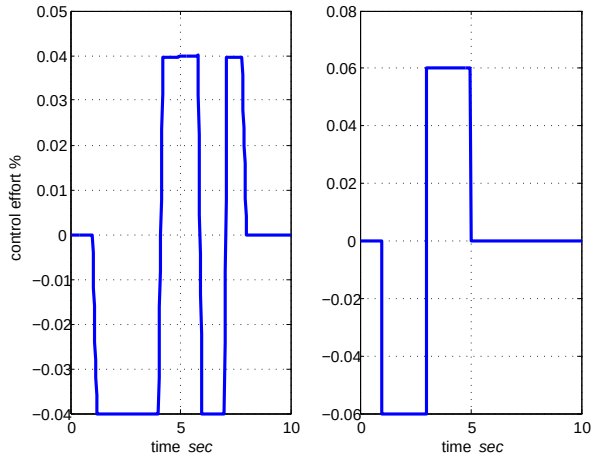

Figure 4.2: Flight test inputs: (1) 3-2-1-1 multistep, (2) doublet.

be determined. Flight test maneuvers such as the multi-step 3211 and doublet input signals shown in Figure 4.2, are widely applied for aircraft system identification (Kaletka and Hamel, 1997).

The multistep 3211 input signal refers to the relative time interval between control reversals. It is used to excite a wide frequency band within a short period of time and is suited for short period motion of stable systems (Kaletka and Hamel, 1997). The signal typically last for 7 secs, then the controls are kept constant until the aircraft is trimmed. A recommended approach for pilot-flown tests is described below:

1. Establish trim for the selected flight condition.
2. Generate a prescribed input in one single control and avoid controls coupling.
3. Let the aircraft respond without further control activity.
4. Trim the aircraft for the next test.

In miniature helicopters, most of the flight tests can be carried out as open-except for yaw dynamics identification due to the active yaw damping system. The stabiliser bar, along side the yaw gyro, is often regarded as feedback systems during the identification process (Mettler, 2003). Depending on the identification objectives, the structure of these systems will have to be considered.

4.1.3 Signal measurement and data analysis

Within the rotorcraft system identification, air data measurement is still a major problem area. In full-scale helicopters, accurate measurements representing the full flight envelope

is seldom obtained. The necessary signals are often reconstructed through kinematic relationships given by the nonlinear differential equations. Such kinematic relationships are the blade flapping dynamics described in Section 2.4.1 since it is virtually impossible to obtain flight data from a miniature helicopter platform (Mettler, 2003).

4.1.4 Description of the X-Cell.60 flight test

The SIMULINK environment is used to develop simulated flight test of the X-Cell.60 miniature helicopter. Unlike real flight data, sensor noise and atmospheric turbulence was inserted through a mixture of additive and multiplicative noise inputs. The control inputs used and recorded for the tests were the stick deflection from the cyclic longitudinal δ_{lon} and cyclic lateral δ_{lat}, collective δ_{col}, yaw pedal δ_r and throttle level δ_t. The following vehicle variables were recorded:

- Euler angles: roll ϕ, pitch θ, yaw ψ
- Angular body rates: roll p, pitch q, yaw r
- Angular body accelerations: roll $\dot{p}$, pitch $\dot{q}$, yaw $\dot{r}$
- Body accelerations: a_x, a_y, a_z
- Body velocities: u, v, w

During the experiment, all the control inputs and system state variables were recorded at 50Hz. The raw flight data for hover, 10 m/s and 20 m/s forward flight was segmented for each experimental run.

4.1.5 Key derivatives

The system identification process requires the verification of performance predictions. This requires a mathematical model representative of the dynamics of the aircraft which is characterized by coefficients or parameters whose numerical values must be determined and vary for various flight conditions. For a miniature helicopter platform, the key derivatives are defined as (Mettler, 2003):

- Decoupled roll and pitch dynamics - Effective rotor time constant τ_e, flapping spring derivatives (M_{a_1},L_{b_1}), effective cyclic input derivatives ($A_{\delta_{lon}}$,$B_{\delta_{lat}}$)
- Coupled roll-pitch dynamics - cross-coupling derivatives (A_{b_1},B_{a_1}), cross-axis cyclic input derivatives ($A_{\delta_{lat}}$,$B_{\delta_{lon}}$)
- Translational longitudinal and lateral dynamics - speed derivatives (X_u and Y_v) and sensor offset (h_{cg})

- Heave dynamics - heave damping (Z_w) and collective input sensitivity ($Z_{\delta_{col}}$)
- Yaw dynamics - yaw damping (N_r), yaw rate feedback (K_r), and tail rotor pedal input sensitivity (N_{δ_r})
- Coupling of heave and yaw dynamics - collective-to-yaw ($N_{\delta_{col}}$) and yaw-to-collective derivatives (Z_r)
- Coupling of heave and pitch dynamics - collective-to-pitch derivative ($M_{\delta_{col}}$)

The design of flight control laws also require continuous availability of these key derivatives to achieve high bandwidth control. It is desirable then to derive the estimates of the above-mentioned parameters directly from flight test data in real-time. Two methods for 'online' parameter estimation are investigated namely: The Delta Method (DM) and the Modified Delta Method (MDM). The DM and MDM are based on model identification using Neural Networks.

4.2 Introduction to Neural Networks

In real-world applications, some nonlinear plants cannot be parametrized linearly (Kenne et al., 2006). In principle, the method of linearization requires the system to operate close to its equilibrium point which may not always be practical for an unstable system with strongly coupled nonlinearities such as a helicopter. The introduction of neural networks as an identification method without *a priori* information, has made neural networks the preferred alternative for flight vehicle identification. Defined as general function approximators, neural networks can approximate a function, provided a solution exists, to any desired accuracy (White and Sofge, 1992).

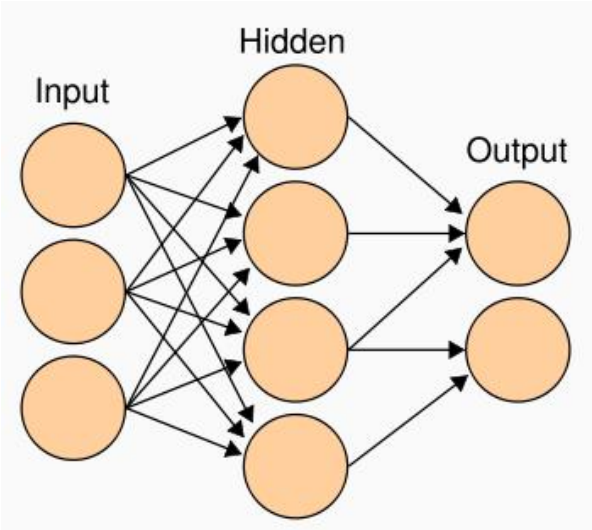

Figure 4.3: Neural Network structure.

A neural network, also known as an Artificial Neural Network (ANN), is a mathematical

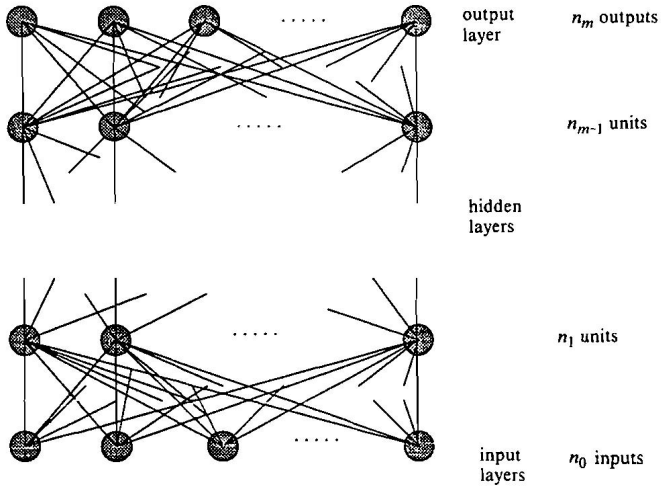

Figure 4.4: A multilayer network architecture

model that tries to simulate aspects of biological neural networks. It consists of an interconnected group of layers namely: the input layer, the hidden or neuron layer and the output layer as shown in Figure 4.3. A neural network is able to change its structure based on the internal or external information that flows through the network during its learning or training phase with minimal user input. This usually occurs through the hidden layer(s) changing the number of neurons needed to model complex relationships between inputs and outputs. This adaptive feature makes neural networks a powerful tool for learning complex mappings generated by dynamical systems with nonlinearities which is the primary goal for the identification of nonlinear dynamic systems (Billings et al., 1992).

Neural networks are often divided into two main groups: static and dynamic networks. The former is known as feedforward neural networks (FFNNs) and the latter as recurrent neural networks (RNNs). The distinctive difference between FFNNs and RNNs is the absence of an output feedback loop in FFNNs. Unlike the general application of FFNNs, the flexibility of RNNs has been shown to be limited because a fixed number of neuron is needed for state-space formulation (Raisinghani, Ghosh and Khubchandani, 1998). In this study, only FFNNs will be considered.

4.3 Feedforward Neural Networks

4.3.1 Network architectures

FFNNs have their neurons arranged in layers in a massive parallel, interconnected network. These hidden layers exist between the input and the output layers. Each neuron of a layer is connected to each neuron of the next layer and each connection is assigned its individual connective weight.

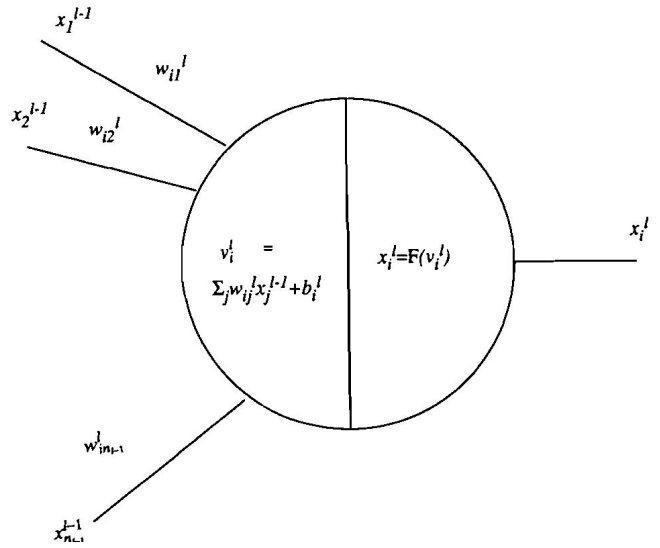

Figure 4.5: Neuron topology

The structure of a multi-layered network is shown in Figure 4.4. The input layers acts as a data holder which distributes inputs to the first hidden layer. The signals flow from the input layer to the output layer. The ith neuron in the lth layer is shown in Figure 4.5.

The neuron has two functions namely: the combining function and the activation function. The combining function produces an activation for the neuron $v_i^l(t)$ defined as

$$v_i^l(t) = \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^{l-i}(t) + b_i^l \tag{4.1}$$

where w_{ij}^l is the weight connection between the jth neuron of the $l-1$ layer and the ith neuron of the lth layer, b_i^l is the threshold of the neuron and n_{l-1} is the number of neurons in the $(l-1)$th layer. The activation function performs a nonlinear transformation to give the output $x_i^l(t)$

$$x_i^l(t) = \mathbf{F}(v_i^l(t)) \tag{4.2}$$

where **F** is the activation or the nonlinear transformation function. $x_i^l(t)$ is the output of the ith neuron of the lth layer for $i = 1, ..., n_l$ and $l = 1, ..., m$. Using these definitions, the mth layer becomes the output layer and input layer can be labeled as zero. Thus n_0 and n_m refer to the inputs and outputs respectively. This is illustrated in Figure 4.4 . Denoting $x_i^0(t)$ and $x_i^m(t)$ as $x_i(t)$ and $\hat{y}_i(t)$ respectively, the ith output node weighted sum can be performed

$$\hat{y}_i(t) = \sum_{j=1}^{n_m-1} w_{ij}^m x_j^{m-1}(t) \tag{4.3}$$

where $x(t) = [x_1(t) \ \dots \ x_{n_0}(t)]^T$ is the input vector to the network. The elements of the input vector are associated to the assigned input nodes of the network. The weights w and bias b

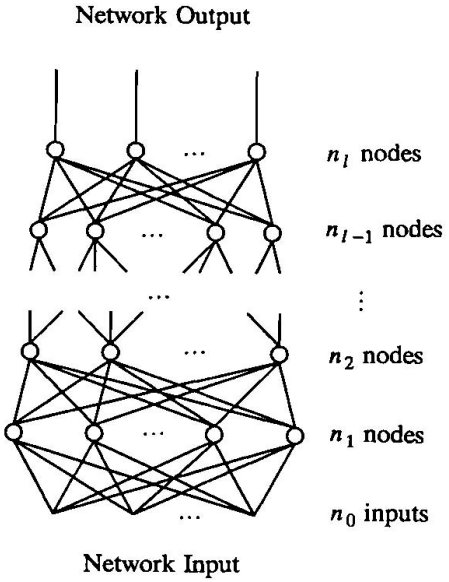

Figure 4.6: Multilayer perceptron structure

are the parameters to be estimated and form elements of Θ, the parameter vector defined as:

$$\Theta = [\theta_1, \theta_2, \dots \ \theta_{n_0}]^T \tag{4.4}$$

The objective of training the neural network model is to determine Θ such that $\hat{y}_i(t)$ is close to the desired output $y(t)$. The residual error is defined as:

$$e(t) = y(t) - \hat{y}(t) \tag{4.5}$$

This approach makes neural networks useful for classification tasks, recognition, vision processing, optimization and function approximation. This is achieved using different network architectures such as: the multilayer perceptron (MLP) and the radial basis function (RBF) networks. The next section discusses the modelling capabilities and the learning algorithms of the above-mentioned network structures (Billings et al., 1992).

4.3.2 Multilayer perceptron

The MLP is a layered network whereby each layer consists of computing nodes. Each node of one layer is fully connected to all the nodes of the adjacent layer. The topology of the MLP is shown in Figure 4.6. The first layer acts as the input layer while the last layer acts as the output layer. All the other layers are seen as hidden layers. The input-output relationship of

the ith node in the kth layer is defined as:

$$z_i^{(k)} = \sum_{j=1}^{n_k-1} \eta_{ij}^{(k)} x_j^{(k-1)} + \mu_i^{(k)} \tag{4.6}$$

$$x_i^{(k)} = \mathbf{F}(z_i^{(k)}) \tag{4.7}$$

where $\eta_{ij}^{(k)}$ and $\mu_i^{(k)}$ are the node connection weights and the threshold respectively. The activation function **F** is often chosen as (Billings et al., 1992):

$$\mathbf{F}(z) = \frac{1}{1+e^{-z}} \tag{4.8}$$

or

$$\mathbf{F}(z) = tanh(z) = \frac{1-e^{-2z}}{1+e^{-2z}} \tag{4.9}$$

Mathematical results (Ljung and Soberstrom, 1983) show that the MLP is a general function approximator which is heavily dependent on the number of hidden nodes or neurons needed to achieve the desired training objective. MLP makes use of learning algorithms such as: batch and recursive identification algorithms which assume the weights have been correctly assigned. Batch and recursive identification algorithms are part of a class of prediction error algorithms which have been derived for MLP for nonlinear system identification. The summarized theory of the prediction error algorithm is described below (Billings and Chen, 1992).

The Prediction Error Algorithm

Assume all the weights and thresholds of the MLP have been arranged into an n_Θ-dimensional vector:

$$\Theta = [\theta_1 \dots \theta_{n_\Theta}]^T \tag{4.10}$$

where

$$n_\Theta = \sum_{i=0}^{l-2}(n_i+1)n_{i+1} + n_{l-1}n_l; \quad n_0 = n_1, \;\; n_l = m \tag{4.11}$$

The input-output equation of the n_1-input and the m-output of MLP is defined:

$$\hat{y}(t,\Theta) = \hat{f}(x(t);\Theta) \tag{4.12}$$

As mentioned earlier, the error residual is defined

$$e(t,\Theta) = y(t) - \hat{y}(t,\Theta) \tag{4.13}$$

The gradient of $\hat{y}(t,\Theta)$ with respect to Θ is a $n_\Theta \times m$ matrix

$$\Psi(t,\Theta) = G(x(t);\Theta) = \left[\frac{d\hat{y}(t,\Theta)}{d\Theta}\right]^T \tag{4.14}$$

The combination of Equation 4.12 and 4.14

$$\begin{bmatrix} \hat{y}(t,\Theta) \\ \Psi(t,\Theta) \end{bmatrix} = \begin{bmatrix} \hat{f}(x(t);\Theta) \\ G(x(t);\Theta) \end{bmatrix} \tag{4.15}$$

the gradient of $\hat{y}(t,\Theta)$ with respect to θ_i is a $1 \times m$ row vector denoted as $\psi_i(t,\Theta)$. Given the weights and threshold of the ith node are arranged such that $1 \leq i \leq q$, where q is the number of nodes, then

$$\Theta = \begin{bmatrix} \Theta_1 \\ \vdots \\ \Theta_q \end{bmatrix} \quad \Psi(t,\Theta) = \begin{bmatrix} \Psi_1(t,\Theta) \\ \vdots \\ \Psi_q(t,\Theta) \end{bmatrix} \tag{4.16}$$

where $\Psi_i(t,\Theta)$, an $n_\Theta \times m$ matrix, is the gradient of $\hat{y}(t,\Theta)$ with respect to Θ_i.

It has been shown that the batch and recursive identification algorithms have the same convergence properties (Ljung and Soberstrom, 1983) whereby the $\hat{\Theta}(t)$ converges to a local minima. The convergence of the widely used backpropagation algorithm (BPA), which is a reduced-order recursive prediction error algorithm, is very slow. Because the MLP is nonlinear in-the-parameters, it typically has a large number of local minimal which may lie at infinity (Billings and Chen, 1992). So when weights fall in these areas, the learning becomes very slow and a long time may elapse before the algorithm may escape. However, the BPA has proven to be popular method and has been used in various disciplines (Kumar et al., 2008; Billings et al., 1992; Ljung and Soberstrom, 1983). A detailed description of the BPA method is given in section 4.5.1 for the parameter estimation of a nonlinear system.

4.3.3 Radial basis function

An alternative to the MLP network is the radial basis function (RBF) networks (Billings et al., 1992). It is described as a two-layer processing structure whereby the hidden layer consists of an array of nodes. Each node contains a parameter function called a *center*. The node computes a nonlinear transformation of the neuron Euclidean distance. This is the distance between the neuron center and network input vector. The architecture of the RBF network is shown in Figure 4.7. The RBF network implements the mapping $\hat{f}_r : \mathbf{R}^n \to \mathbf{R}$ according to

$$\hat{f}_r(\mathbf{x}) = \eta + \sum_{j=1}^{n_h} \eta_{ij}\phi\left(\|\mathbf{x} - \mathbf{c}_j\|, \rho_j\right), \quad 1 \leq i \leq m \tag{4.17}$$

where $\mathbf{x} \in \mathbf{R}_n$ is the input vector, $\phi(\cdot)$ is the given function from the $\mathbf{R}^n$ to $\mathbf{R}$, $\|\cdot\|$ denotes the Euclidean norm, η, $0 \leq i \leq n_h$, are the weights or parameters, $\mathbf{c}_j \in \mathbf{R}^n$, $0 \leq i \leq n_h$, are

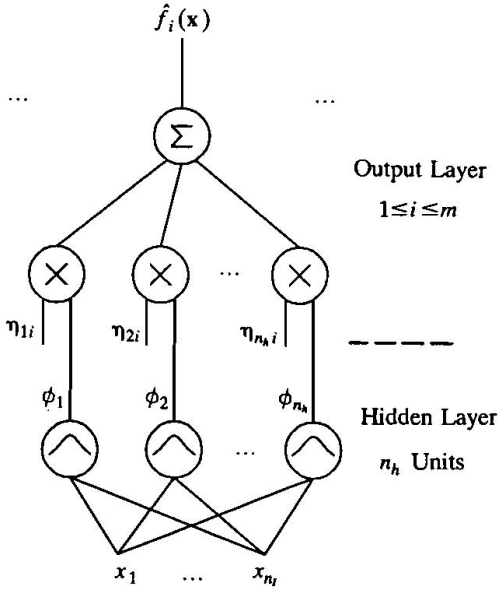

Figure 4.7: Radial basis function network structure

the RBF centers and n_h are the number centers. ρ_j is the centers width. Each node performs a nonlinear transformation $\phi(\cdot)$ of the input vector $\mathbf{x}$ based on the Euclidean distance to the node center $\mathbf{c}_j$. Computation of weights η is achieved by performing a pseudo-inverse a output vector $\hat{f}_r(\mathbf{x})$. It has been shown that the choice of the nonlinear transformation function $\phi(\cdot)$ is not crucial to the performance of the RBF network (Chen et al., 1991; Billings and Chen, 1992). Typical choices of $\phi(\cdot)$ are the thin-plate spline function

$$\phi(z, 1) = z^2 log(z) \tag{4.18}$$

and the Gaussian function, shown in Figure 4.8

$$\phi(z, \rho) = exp(-z^2/\rho^2) \tag{4.19}$$

where ρ is a real constant. Two other common choices of $\phi(\cdot)$ are the multiquadratic function

$$\phi(z, \rho) = \left(z^2 + \rho^2\right)^{1/2} \tag{4.20}$$

and the inverse multiquadric function

$$\phi(z, \rho) = \left(z^2 + \rho^2\right)^{-1/2} \tag{4.21}$$

It is assumed that $\phi(\cdot)$ is continuous and bounded (Park and Sandberg, 1991). This is only valid for the multiquadratic and Gaussian nonlinear functions. For this study the Gaussian nonlinear function is used for the RBF centers. Similarities in the approximation capabilities of the RBF and MLP (two-layer) have been proved (Billings and Chen, 1992). Just as in the case of MLP network, the RBF network can be estimated using the prediction error estima-

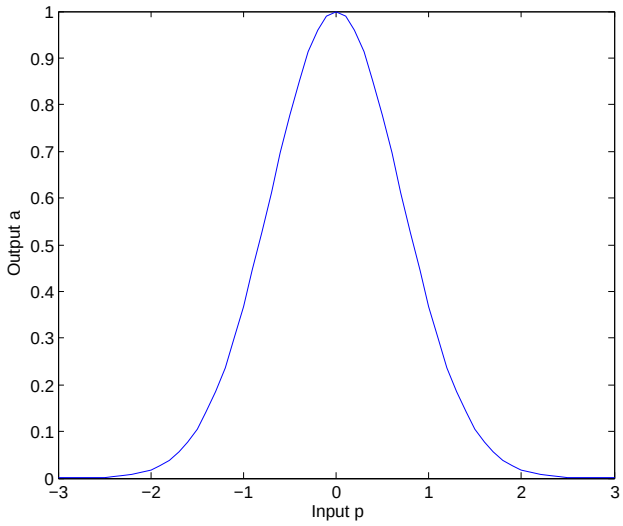

Figure 4.8: RBF neuron with Gaussian function

tion method described above. However, in order to exploit the structure of the RBF network, a linear learning rule was developed (Chen et al., 1991; Song et al., 2005).

Unlike the MLP architecture, the RBF is linear with respect to the network weights. This makes the learning of the network weights a linear problem thus resulting in a lower convergence time. The strategy is to select some data points as centers for weight learning using the least squares (LS) method. This, however introduced instability in the learning scheme and increased convergence time. A more robust approach was developed called the orthogonal least squares (OLS) method which made use of a subset selection method (Chen et al., 1991). A detailed description of the OLS for parameter estimation is given in section 4.5.2.

4.4 Neural Network Model Identification

Model identification is defined as an extraction process by which the system dynamics is approximated through the parametrization of a representative mathematical model using measured data. The identification of nonlinear dynamical systems has shown success in the presence of parameter uncertainties and measurement noise. Frequency-domain identification methods such as CIFER (Mettler et al., 1999; Mettler, Kanade and Tischler, 2000; Mettler, Kanade, Tischler and Messner, 2000; Tischler, 1995) have been used to develop accurate parameterized models around specific trim conditions. Time domain identification methods have also been applied such as: the output error methods (Jategaonkar, 2008), equation error methods (Mehra et al., 1974; Paris and Bonner, 2004) and extended Kalman filtering methods (EKF) (Jun et al., 1998). However, these methods require *a priori* information to estimate all the required system parameters and fail to approximate the system dynamics globally (Kumar et al., 2008; Suresh et al., 1995).

Due to their approximation capabilities and learning complex mappings, neural networks

have been used as an alternative to the identification of nonlinear dynamic systems (Songwu and Basar, 1998; Kumar et al., 2003; Singh and Ghosh, 2007; Raisinghani, Ghosh, and Kalra, 1998; Raisinghani, Ghosh and Khubchandani, 1998; Raisinghani and Ghosh, 2001). One basic requirement in using neural networks architectures such as MLP or RBF is for these architectures to correctly model the behaviour of the dynamic system under investigation. The neural network identification problem is to obtain a set of neuron weights values such that the network's response, based on a specified input set, adequately predicts the behaviour of the given dynamic system (Polycarpou and Ioannou, 1991).

The identification of nonlinear systems of the form

$$\dot{x} = f(x) + g(x)u \tag{4.22}$$

where $u \in \Re$ is the input, $x \in \Re^n$ is the state (obtained from measurements). The identification problem consists of choosing the appropriate identification model such that the unknown nonlinearities of $f(x)$ and $g(x)$ are parametrized by the static neural network with outputs $\hat{f}(x, \theta_f)$ and $\hat{g}(x, \theta_g)$ respectively, where $\theta_f \in R^{n_f}$, $\theta_g \in R^{n_g}$ are the adjustable weights and n_f, n_g denote the number of weights with respect to network approximation of f and g. Equation 4.22 can be expressed in the compact form (Polycarpou and Ioannou, 1991):

$$\dot{x} = \hat{f}(x, \theta_f^*) + \hat{g}(x, \theta_g^*)u + \nu(t) \tag{4.23}$$

where θ_f^* and θ_g^* denote the optimal weights values in the approximation of $f(x)$ and $g(x)$ respectively such that

$$\theta_f^* := arg_{\theta_f} min \left[f(x) - \hat{f}(x, \theta_f) \right] \tag{4.24}$$

$$\theta_g^* := arg_{\theta_g} min \left[g(x) - \hat{g}(x, \theta_g) \right] \tag{4.25}$$

and $\nu(t)$ is the *modelling error* defined as (Polycarpou and Ioannou, 1991):

$$\nu(t) = \left[f(x(t)) - \hat{f}(x, \theta_f^*) \right] + \left[g(x(t)) - \hat{g}(x, \theta_g^*) \right] \tag{4.26}$$

4.4.1 Model validation

Various validation methods have been developed for the identification of the nonlinear systems in the presence of noise and input delays (Billings et al., 1992). The neural network modelling of a system is deemed adequate only if the residuals or prediction errors $e(t)$, computed from the testing set, are unpredictable from all linear and nonlinear combinations of past inputs and outputs (Billings et al., 1992). An incorrect prediction of the system output could be caused by the following factors: incorrect input node assignment, noisy data and insufficient hidden nodes. Although the network will be able to train and minimize the cost

function, these factors are clearly illustrated by simulating the network with a different input set (testing set).

Effect of noise

Real systems inherit a certain level of noise through their instrumentation or from atmospheric disturbance which cannot be ignored. Noisy data (both measurement and process noise) is known to introduce severe bias in the network input-output data. While a good prediction is provided over the training data, an incorrect prediction often arises over the testing data concluding the inability of the network to model the system underlying dynamics thereby limiting the network performance.

Network complexity

The performance of the network is also directly linked to the size of its hidden layers and nodes. It has been shown that increasing network complexity does not necessarily lead to better network performance but often causes the model to become a high dimensional curve fit to the training data (Billings et al., 1992; Billings and Chen, 1992). This is referred to as the generalization for neural networks. This overfitting can be avoided by developing a network model whereby the number of neurons increase incrementally until the validation criteria has been met. No other optimization methods has been applied to minimize network size over and above the mentioned technique.

4.4.2 Comparison of RBF and MLP

To illustrate these validation mechanisms, a RBFN and MLPN are trained and validated using a sine-based function. Two cases of ideal and noisy data is investigated. A comparison of the RBF and MLP network performance is also described. Given an input vector (Kar and Behera, 2009):

$$x = [x_1 \quad x_2]^T$$

where x_1 and x_2 are the state variables. The nonlinear function $f(\mathbf{x})$ is described:

$$f(x) = 4\left(\frac{sin(4\pi x_1)}{\pi x_1}\right)\left(\frac{sin(\pi x_2)}{\pi x_2}\right)^2$$

the following variable ranges have been specified:

- x_1 = 0.01 to 50
- x_2 = 0.01 to 50

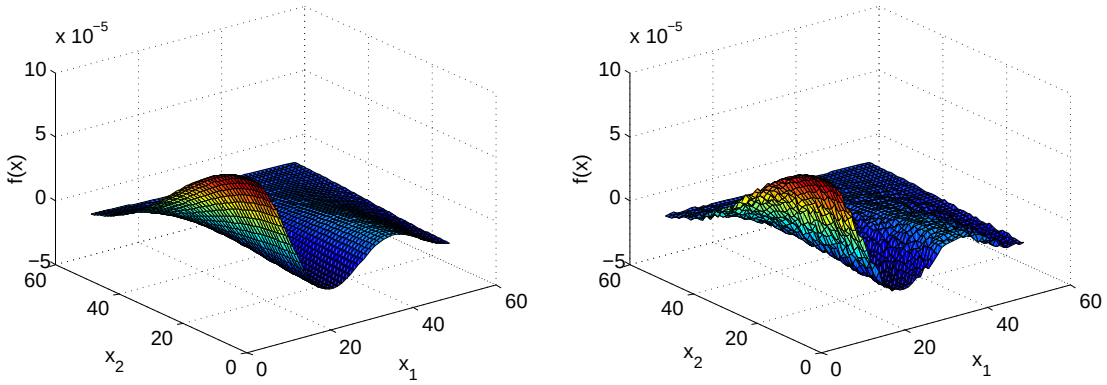

Figure 4.9: Surface plot of $f(x)$: (1) Noise-free, (2) 10% noise.

An increment of 0.1 is used for both state variables resulting in 500 samples. Figure 4.9 shows the surface plots of $f(x)$ to be identified using neural networks. It also shows the effect noisy state variables have on the surface plot $f(x)$. This is used to evaluate the effect of noise on network performance.

The training and testing sets are created by splitting the input-output data.

- Samples 1-250 for training
- Samples 251-500 for testing

A goal value of 1e-12 was used for both networks for this analysis. The RBF network spread constant was specified at 0.8326 for all its neurons. The MLP network specifications are given below:

1. Nodes structure - (12-20-1) (input-hidden-output)
2. Learning rate - 0.01
3. Minimum gradient - 1e-9

Table 4.1: RBF and MLP validation results

	Noise-free		**10% noise**	
	RBF	MLP	RBF	MLP
Prediction error	1.31e-6	2.33e-4	2.38e-6	0.0048
No. of neurons	12	20	12	20
Convergence time(s)	0.94	4.88	5.95	6.33

Table 4.1 compares the network performance of the RBF and MLP networks. The robustness of the RBF algorithm can be seen as the prediction error varies slightly with a large increase in noise. The MLP results are as expected with a large increase in error as system noise increases. Since the MLP network structure must be specified prior to learning, MLP network

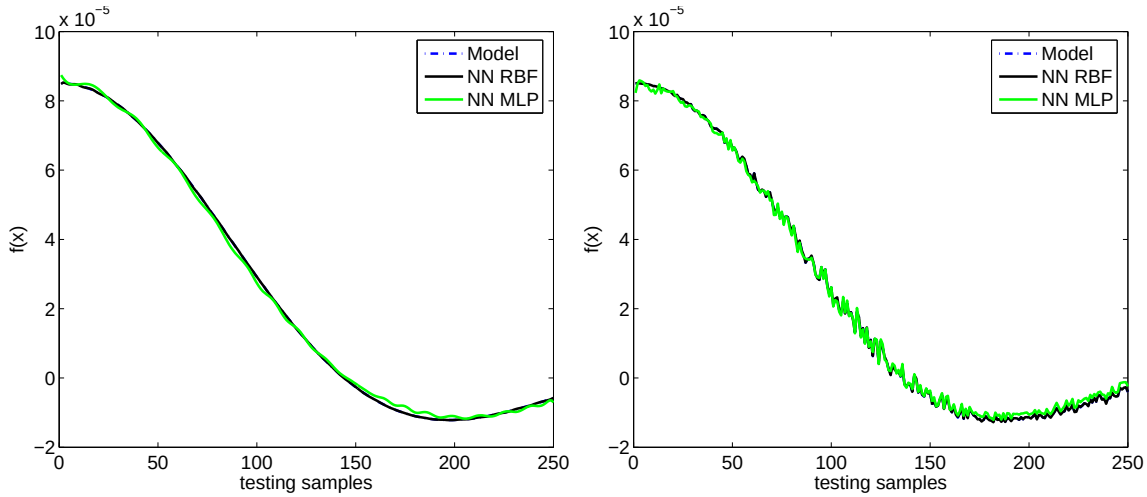

Figure 4.10: Network training results: (1) Noise free, (2) 10% noise.

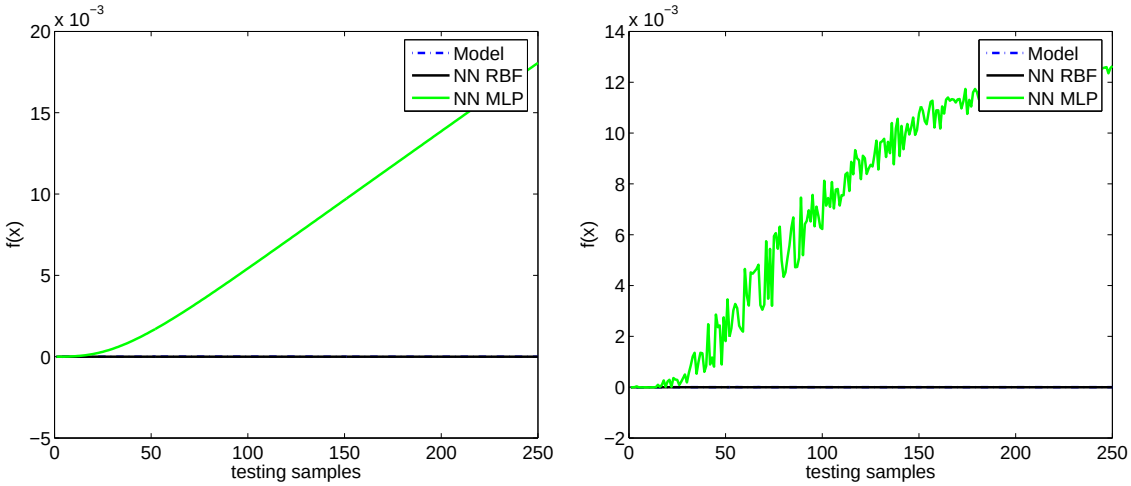

Figure 4.11: Network validation results: (1) Noise free, (2) 10% noise.

complexity is unaffected by modelling errors or uncertainties. RBF network increases in complexity as it tries to minimize the prediction error one neuron at a time. In this case however, the same number of neurons (12) resulted in the noise-free and 10% noise case. This shows the RBFN robustness to noise. Another measure of performance is the convergence time. As expected, the MLP network showed slow convergence in both the noise-free and 10% noise cases with 4.88 and 6.33 seconds respectively. Unlike the latter, RBF network showed a significant increase in convergence time from 0.98 to 5.95 seconds. Although the size of the network is unchanged, the low signal-to-noise ratio causes the weight computation to increase as the prediction error is minimized.

Figure 4.10 shows that both networks trained adequately in the noise-free and 10% noise cases. The complex error surface produced by the MLP network was clearly visible in the noise-free case and resulted in the network inability to approximate at the extremities of the sampled data (in this case the first sample). This result can also be found in (Kumar et al., 2008). The prediction of the function $f(x)$ underlying dynamics was investigated through simulating both networks using the testing set shown in Figure 4.11. It is apparent that the

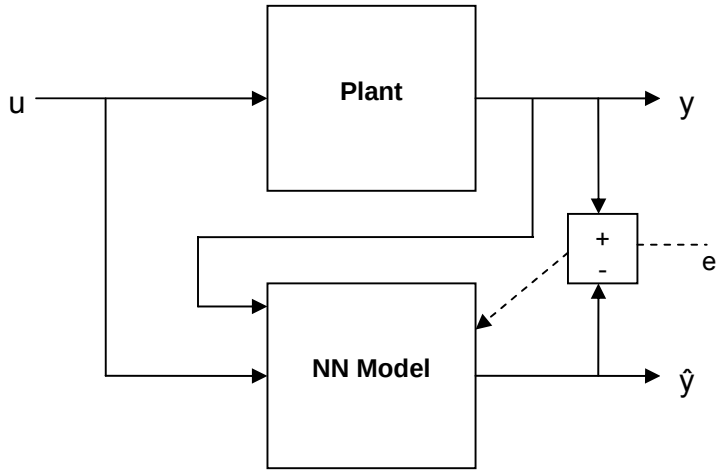

Figure 4.12: NN model identification structure

MLP network resulted in large inaccuracies in both the noise-free and 10% noise cases. The RBF network, although not clear due to scaling, showed good prediction capabilities and robustness to noise.

It can be concluded that the RBF network is well suited for online identification. Its robustness to noise, evolving network structure and the ability to identify the underlying dynamics provides a foundation for the network to adapt to changes that occur as the helicopter moves from one flight condition to another. The decrease in convergence time due to the effect of noise can be circumvented through the proper choice of the spread constant and goal value.

4.4.3 Online identification

The online identification using neural networks has been studied (Jafari et al., 2007; Singh, 2005). Singh (2005) used neural networks to identify and compensate for the modelling errors introduced by the inverse modelling of the system dynamics. The following modelling errors were identified: partial actuator failure, weight change and center of gravity change. Longitudinal flight control was then developed using networks for flight velocity and flight path angle control.

The model structure for online identification is shown in Figure 4.12. Using the flight instrumentation described in Section 4.1.1, measured plant data (both input u and output y) is supplied to the neural network for the identification of the underlying system dynamics. The neural network output is given as $\hat{y}$. The residual error e is minimized by updating neuron weights. One distinct drawback using feedforward neural networks for online identification, is the network inability to obtain a global mapping from a single input-output pattern without 'forgetting' the underlying dynamics. A moving window algorithm was introduced

to improve the network learning performance (Savran et al., 2006; Jafari et al., 2007). Savran et al. (2006) describe it as a first-in-first-out stack to store a short history of the input output data.

4.5 Aerodynamic Parameter Estimation

The development of flight control systems requires the mathematical model of the helicopter to be a close match to the real system. Preliminary parameter estimation methods such as: wind tunnel testing and computational fluid dynamics, remain valuable but their prediction of cross-coupling effects are ineffective (Jategaonkar, 2008). The estimation of stability and control derivatives is based on stringent specifications for flight control and in-flight simulator design. Frequency domain identification methods such as CIFER (Mettler et al., 1999; Mettler, Kanade and Tischler, 2000; Mettler, Kanade, Tischler and Messner, 2000; Tischler, 1995) have been used to develop accurate parameterized models around specific trim conditions. Morelli (2000) made use of a recursive Fourier transform algorithm to generate a linear dynamic state-space model for real-time parameter estimation.

Aerodynamic parameter estimation using neural network, FFNNs in particular, has been applied extensively (Raisinghani, Ghosh, and Kalra, 1998; Raisinghani, Ghosh and Khubchandani, 1998; Raisinghani and Ghosh, 2001; Kumar et al., 2008). The methods do not require a priori information of the system dynamics and the aerodynamic parameters can be extracted indirectly from the flight data. Although neural networks have been primarily used for model identification, methods such as the delta method and modified delta method using RBFN, have been able to estimate indirectly stability and control derivatives both for simulated ideal and noisy data (Kumar et al., 2008).

Noise and data information content are two main problems that occur during online/real-time parameter estimation (Morelli, 2000). The objective is to design a technique which is insensitive to noise and still responds rapidly to sudden changes in the system dynamics. Different neural network architectures such as: MLP and RBF, have been used to overcome these problems and act as an alternative approach to aerodynamic parameter estimation.

4.5.1 MLP-based estimation

The recursive prediction error estimator has been used for the online identification of parameters using helicopter flight data (Suresh et al., 1995; Chen and Billings, 1989). This is derived by minimizing the discrepancy between the measured output and the predicted output by using a candidate model (prediction error).

The backpropagation algorithm

The backpropagation algorithm (BPA), a class of recursive prediction error algorithm, is defined using the time-varying version of the extended model in Equation 4.15

$$\begin{bmatrix} \hat{y}(t,\Theta) \\ \Psi(t,\Theta) \end{bmatrix} = \begin{bmatrix} \hat{f}(x(t);\Theta);\ \hat{\Theta}(t-1) \\ G(x(t);\Theta);\ \hat{\Theta}(t-1) \end{bmatrix} \tag{4.27}$$

where $\hat{\Theta}(t-1)$ denotes the estimate of Θ at t. The approximate prediction error is defined

$$e(t) = y(t) - \hat{y}(t) \tag{4.28}$$

the steepest (gradient) descent algorithm can be defined (Billings and Chen, 1992)

$$\Delta(t) = \alpha_m \Delta(t-1) + \alpha_g \Psi(t) e(t) \tag{4.29}$$

with

$$P(t) = \left[P(t-1) - P(t-1)\Psi(t)(\lambda I + \Psi^T(t)P(t-1)\Psi(t))^{-1}\Psi^T(t)P(t-1)\right]/\lambda \tag{4.30}$$

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + P(t)\Delta(t) \tag{4.31}$$

where α_g and α_m are the learning rate parameter and momentum constant respectively and λ is the forgetting factor. the smoothed stochastic gradient algorithm

$$\left.\begin{array}{c} \sigma_i(t) = \alpha_m \sigma_i(t-1) + \alpha_g \Psi_i(t) e(t) \\ \hat{\theta}_i(t) = \hat{\theta}_i(t-1) + \sigma_i(t) \end{array}\right\} 1 \leq i \leq n_\theta \tag{4.32}$$

The quantity $\Psi_i(t)e(t)$ corresponds to the negative gradient of $e^T(t)e(t)/2$ with respect to θ_i and is stochastic in nature. σ_i is therefore stochastic gradient. Given an input-output relationship of the ith node in the kth layer is defined as

$$z_i^{(k)} = \sum_{j=1}^{n_k - 1} \eta_{ij}^{(k)} x_j^{(k-1)} + \mu_i^{(k)} \tag{4.33}$$

$$x_i^{(k)} = \mathbf{F}(z_i^{(k)}) \tag{4.34}$$

where $\eta_{ij}^{(k)}$ and $\mu_i^{(k)}$ are the node connection weights and the threshold respectively, the BPA for MLP training can then be formulated (see section 4.3.2)

$$\left.\begin{array}{c} \eta_{ij}^{(k)}(t) = \eta_{ij}^{(k)}(t-1) + \Delta\eta_{ij}^{(k)}(t) \\ \mu_i^{(k)}(t) = \mu_i^{(k)}(t-1) + \Delta\mu_{ij}^{(k)}(t) \end{array}\right\} \tag{4.35}$$

with

$$\left.\begin{array}{c}\Delta\eta_{ij}^{(k)}(t) = \alpha_m \Delta\eta_{ij}^{(k)}(t-1) + \alpha_g \delta_i^{(k)}(t) x_j^{(k-1)}(t) \\ \Delta\mu_i^{(k)}(t) = \alpha_m \Delta\mu_i^{(k)}(t-1) + \alpha_g \delta_i^{(k)}(t)\end{array}\right\} \tag{4.36}$$

and

$$\delta_i^{(l)} = a'(z_i^{(l)}(t))(y_i(t) - \hat{y}_i(t)) \tag{4.37}$$

$$\delta_i^{(k)} = a'(z_i^{(l)}(t)) \sum_s \delta_s^{(k+1)}(t) \eta_{si}^{(k+1)}(t-1), \quad k = l-1, \ldots, 2, 1 \tag{4.38}$$

where $a'(z)$ is the derivative of $a(z)$.

4.5.2 RBF-based estimation

Parameter estimation using RBF-based methods have been applied (Kumar et al., 2008; Kenne et al., 2006; Jafari et al., 2007; Park and Sandberg, 1991; Song et al., 2005). It was shown that the linear-in-the-parameter RBF structure was able to approximate strongly coupled aerodynamic parameters (Kumar et al., 2008). The application of adaptive RBF-based neural networks was investigated by Jafari et al. (2007). A growing and pruning radial basis function (GAP-RBF) and minimal resource allocation network (MRAN) were combined with an unscented Kalman filter (UKF) training algorithm for online parameter estimation of the system dynamics. The RBF network is discussed in more detail in section 4.3.3.

Orthogonal Least Square Algorithm

The parametrization of a regression model requires that there be one basis function per data point. This approach quickly becomes computationally costly as new data points are made available. The Orthogonal Least Squares (OLS) algorithm is used as a systematic approach to radial basis centers selection (Chen et al., 1991). This method is employed as a forward regression procedure whereby a fixed center corresponds to a given regressor in a linear regression model. The latter is defined as:

$$d(t) = \sum_{i=1}^{M} p_i(t)\theta_i + e(t) \tag{4.39}$$

where $d(t)$ is the desired output (also called the dependent variable), θ_i are the parameters, and $p_i(t)$ are known as the regressors which are fixed functions of $\mathbf{x}(t)$ such as:

$$p_i(t) = p_i(\mathbf{x}(t)) \tag{4.40}$$

The error signal $e(t)$ is assumed to be uncorrelated with the regressors $p_i(t)$. Assuming each regressor to be a constant term, the problem of fixed RBF center $\mathbf{c}_i$ selection with a given nonlinearity $\phi(\cdot)$ corresponds to selecting a suitable set of regressors such that $e(t)$ can be proved to be uncorrelated to the system regressors. Given a linear regression matrix system:

$$\mathbf{d} = \mathbf{P}\Theta + \mathbf{E} \tag{4.41}$$

where

$$\mathbf{P} = [\mathbf{p}_1 \cdots \mathbf{p}_M], \quad \mathbf{p}_i = [p_i(1) \cdots p_i(N)]^T, \quad 1 \leq i \leq M \tag{4.42}$$

$$\Theta = [\theta_1 \cdots \theta_M]^T \tag{4.43}$$

$$\mathbf{E} = [e(t) \cdots e(N)]^T \tag{4.44}$$

The regressor $\mathbf{p}_i$ forms a set of basis vectors where the least squares (LS) solution $\hat{\Theta}$ satisfies the condition that $\mathbf{P}\hat{\Theta}$ be the projection of $\mathbf{d}$ onto space spanned by these basis vectors. The square of the projection $\mathbf{P}\hat{\Theta}$ is part of the desired output energy produced by the regressors. Since regressors are generally correlated such as velocity and thrust in airplanes, the output energy due to one regressor is not clearly defined.

The Orthogonal Least Squares method transforms the set $\mathbf{p}_i$ into a set of orthogonal basis vectors, and thus enables the computation of an individual basis vector (regressor) contribution to the desired output energy. The regression matrix $\mathbf{P}$ can be decomposed into

$$\mathbf{P} = \mathbf{WA} \tag{4.45}$$

where $\mathbf{A}$ is an $M \times M$ triangular matrix given as:

$$A = \begin{bmatrix} 1 & \alpha_{12} & \alpha_{13} & \cdots & \alpha_{1M} \\ 0 & 1 & \alpha_{23} & \cdots & \alpha_{2M} \\ 0 & 0 & 1 & 0 & 0 \\ \vdots & & \ddots & 0 & 0 \\ & & & 1 & \alpha_{M-1M} \\ 0 & \cdots & 0 & 0 & 1 \end{bmatrix}$$

and $\mathbf{W}$ is an $N \times M$ matrix with orthogonal columns $\mathbf{w}_i$ such that

$$\mathbf{W}^T\mathbf{W} = \mathbf{H} \tag{4.46}$$

where H is a diagonal matrix with elements h_i:

$$\mathbf{h}_i = \mathbf{w}_i^T \mathbf{w}_i = \sum_{t=1}^{N} w_i(t) w_i(t), \quad 1 \leq i \leq M \tag{4.47}$$

Since the same space is spanned by the orthogonal vector $\mathbf{w}_i$ and the regressors $\mathbf{p}_i$, Eq.4.41 can be rewritten as:

$$\mathbf{d} = \mathbf{W}\mathbf{g} + \mathbf{E} \tag{4.48}$$

The orthogonal (LS) solution $\hat{g}$ is given as:

$$\hat{\mathbf{g}} = \mathbf{H}^{-1}\mathbf{W}^T\mathbf{d} \tag{4.49}$$

or

$$\hat{\mathbf{g}}_i = \mathbf{w}_i^T\mathbf{d}/(\mathbf{w}_i^T\mathbf{w}_i), \quad 1 \leq i \leq M \tag{4.50}$$

The orthogonal decomposition of **P** can be obtained using the Gram-Schmidt algorithm (Chen et al., 1991). This method computes the one column of matrix A at a time and orthogonalizes P as follows: at the kth stage make the kth column orthogonal to each of the $k-1$ previously orthogonalized columns and repeat the operation for $k = 2, \cdots, M$. This procedure is represented as:

$$\mathbf{w}_1 = \mathbf{p}_1$$

$$\alpha_{ik} = \mathbf{w}_i^T\mathbf{p}_k/(\mathbf{w}_i^T\mathbf{w}_i), \quad 1 \leq i \leq k$$

$$\mathbf{w}_k = \mathbf{p}_k - \sum_{i=1}^{k-1} \alpha_{ik}\mathbf{w}_i$$

Because $\mathbf{w}_i$ and $\mathbf{w}_j$ are orthogonal for $i \neq j$, the sum squares or energy of $d(t)$ is

$$\mathbf{d}^T\mathbf{d} = \sum_{i=1}^{M} g_i^2 \mathbf{w}_i^T\mathbf{w}_i + \mathbf{E}^T\mathbf{E} \tag{4.51}$$

If **d** is the desired output vector after the mean has been removed, then the variance of d(t) is given by

$$N^{-1}\mathbf{d}^T\mathbf{d} = N^{-1}\sum_{i=1}^{M} g_i^2 \mathbf{w}_i^T\mathbf{w}_i + N^{-1}\mathbf{E}^T\mathbf{E} \tag{4.52}$$

It can be noticed that the desired output variance is divided between regressor-induced variance and unexplained variance of *d(t)*. Thus $g_i^2\mathbf{w}_i^T\mathbf{w}_i/N$ is the increment to the explained desired output introduced by $\mathbf{w}_i$. Dividing Eq. 4.52 by $N^{-1}\mathbf{d}^T\mathbf{d}$, the explained error can be computed:

$$\mathbf{E}^T\mathbf{E}/\mathbf{d}^T\mathbf{d} = 1 - \frac{1}{\mathbf{d}^T\mathbf{d}} g_i^2 \mathbf{w}_i^T\mathbf{w}_i \tag{4.53}$$

So from Eq.4.53 it can be seen that a maximum value of the ratio $g_i^2 \mathbf{w}_i^T \mathbf{w}_i / \mathbf{d}^T \mathbf{d}$ will produce a minimum unexplained error. This approach offers an effective means of seeking a subset of significant regressors $\mathbf{w}_i$ that will minimize the unexplained error in the desired output. Therefore given an error reduction ratio:

$$[err]_i = g_i^2 \mathbf{w}_i^T \mathbf{w}_i / \mathbf{d}^T \mathbf{d} \tag{4.54}$$

The selection of regressors is such that

$$1 - \sum_{j=1}^{M_s} [err]_j < \rho \tag{4.55}$$

where $0 < \rho < 1$ is a chosen tolerance and M_s is a subset model containing the significant regressors. The regressor selection procedure can be summarized as follows: At the first step, for $1 \leq i \leq M$

$$\mathbf{w}_1 = \mathbf{p}_i$$

$$g_1^{(i)} = (\mathbf{w}_1^{(i)})^T \mathbf{d} / ((\mathbf{w}_1^{(i)})^T \mathbf{w}_1^{(i)})$$

$$[err]_1^{(i)} = (g_1^{(i)})^2 (\mathbf{w}_1^{(i)})^T \mathbf{w}_1^{(i)} / (\mathbf{d}^T \mathbf{d})$$

Find

$$[err]_1^{(i_1)} = max[err]_1^{(i)}$$

and select

$$\mathbf{w}_1 = \mathbf{w}_1^{(i_1)} = \mathbf{p}_{i_1}$$

At the *k*th step, where $k \geq 2$

$$\alpha_{jk}^{(i)} = \mathbf{w}_j^T \mathbf{p}_i / (\mathbf{w}_j^T \mathbf{w}_j)$$

$$\mathbf{w}_k^{(i)} = p_i - \sum_{j=1}^{k-1} \alpha_{jk}^{(i)} \mathbf{w}_j$$

$$g_k^{(i)} = (\mathbf{w}_k^{(i)})^T \mathbf{d} / ((\mathbf{w}_k^{(i)})^T \mathbf{w}_k^{(i)})$$

$$[err]_k^{(i)} = (g_k^{(i)})^2 (\mathbf{w}_k^{(i)})^T \mathbf{w}_k^{(i)} / (\mathbf{d}^T \mathbf{d})$$

find

$$[err]_k^{(i_1)} = max[err]_k^{(i)} \quad 1 \leq i \leq M,\ i \neq i_1$$

and select

$$\mathbf{w}_k = \mathbf{w}_k^{(i_k)} = p_{i_k} - \sum_{j=1}^{k-1} \alpha_{jk}^{(i_k)} \mathbf{w}_j$$

The procedure is terminated at the M_sth step when

$$1 - \sum_{j=1}^{M_s} [err]_j < \rho$$

The vector matrix of $\mathbf{w}_k^{(i_k)}$, contained in the subset M_s, embodies the desired output energy

hence representative of the system dynamics. Within a Neural network context, these represent the number of neuron centers $\mathbf{c}_i$ is the hidden layers such that the output layer 'predicts' the desired output.

4.5.3 Delta method algorithm

The Delta Method (DM) is a central different technique based on the understanding that a stability/control derivative is the change in the aerodynamic force or moment coefficient caused by a small variation in one of the motion/control variables about its nominal value while all the other variables are held constant (Raisinghani, Ghosh, and Kalra, 1998). This is defined as:

$$\frac{dy}{dx}_{x=x_i} = \frac{\hat{f}(x_1, \dots x_i + h, \dots, x_n) - \hat{f}(x_1, \dots x_i - h, \dots, x_n)}{2h} \tag{4.56}$$

where $\hat{f}(x)$ is the approximation of the function identified by the neural network. y and x_i are the output and input set used for function approximation. Equation 4.56 is the central difference approximation of the derivative $\frac{dy}{dx}$ at point $x = x_i$. The following procedures must be adhered to calculate the derivative:

1. The network is trained with a set of input-output values (training set) to create the approximate function $\hat{f}(x)$.
2. Network validation is achieved using a testing set with the output residuals falling below a set goal value.
3. The input file of the training set is 'positive' modified by perturbing an input x_i with $+h$.
4. This input file is then used to generate the function approximation $\hat{f}(x)_{\Delta=+h}$.
5. Same process is repeated with a 'negative' modified input file to produce the function approximation $\hat{f}(x)_{\Delta=-h}$
6. Equation 4.56 can then be used to compute the derivative of interest.

Example

Considering an input-output set which includes the x-axis acceleration a_x and the forward velocity u. The computation of derivative X_u can be defined:

$$X_u = \frac{dX}{du}_{x_i=u} = \frac{X^+ - X^-}{2\Delta u} \tag{4.57}$$

where X^+ and X^- are the positive and negative modified output and Δu is the perturbation in network input variable u.

4.5.4 Modified delta method algorithm

The Modified Delta Method (MDM) is based on interpreting that a stability/control derivative is the ratio between the variation of the aerodynamic coefficient and the variation of the motion/control variable while the variation in other motion/control variables are identically zero. Similarly to DM, the computation of the derivative is defined (Raisinghani and Ghosh, 2001):

$$\frac{dy}{dx}_{x=x_i} = \frac{\hat{f}(\Delta x_1, \ldots \Delta x_i \times 0, \ldots, \Delta x_n)}{\Delta x_i} \tag{4.58}$$

where $f(\Delta x)$ is the approximation function using the differential variation of x. The following procedures have been defined:

1. The network is trained with a set of differential input-output values (training set) to create the approximate function $\hat{f}(\Delta x)$.
2. Network validation is achieved using a testing set with the output residuals falling below a set goal value.
3. The input file of the training set is modified by changing all inputs to zero while leaving input Δx_i unchanged.
4. This input file is then used to generate the function approximation $\hat{f}(\Delta x_i)$.
5. Equation 4.58 can then be used to compute the derivative of interest.

Example

Considering an input-output set which includes the x-axis acceleration a_x and the forward velocity u. The computation of derivative X_u can be defined:

$$X_u = \frac{dX}{du}_{x_i=u\neq 0} = \frac{\Delta X^{(\Delta u)}}{\Delta u} \tag{4.59}$$

where $\Delta X^{(\Delta u)}$ is the modified output with respect to Δu and Δu is the differential input variable of u.

4.5.5 Parameter statistics

In order to evaluate the performance of the estimation methods described, statistical methods have been employed. A detailed description of the parameter statistics is given in Appendix B on page 184.

4.5.6 Online estimation model

The performance of the online estimation model depends to a large degree on the successful identification of the external forces and moments acting on the flight vehicle. These can be classified as: aerodynamic, inertial, gravitational and propulsive. Because aerodynamic forces (X,Y,Z) and moments (L,M,N) cannot be measured directly, aerodynamic modelling followed by parameter estimation determines these parameters with respect to the related measurements such as accelerations, angular rates, linear translations and control inputs. The key derivatives defined within the identification model structure of a 6-DOF helicopter are given (Lorenz and Chowdhary, 2005; Mettler et al., 1999):

$$a_x = \dot{u}(t) = X_u u(t) + -g\theta + X_{a_1} a_1(t) + X_{b_1} b_1(t)$$

$$a_y = \dot{v}(t) = Y_v v(t) + g\phi + Y_{a_1} a_1(t) + Y_{b_1} b_1(t)$$

$$a_z = \dot{w}(t) = Z_w w(t) + Z_{a_1} a_1(t) + Z_{b_1} b_1(t) + Z_{\delta_{col}} \delta_{col}(t)$$

$$\dot{p}(t) = L_u u(t) + L_v v(t) + L_{a_1} a_1(t) + L_{b_1} b_1(t)$$

$$\dot{q}(t) = M_u u(t) + M_v v(t) + M_{a_1} a_1(t) + M_{b_1} b_1(t) + M_{\delta_{col}} \delta_{col}(t)$$

$$\dot{r}(t) = N_r r(t) + N_{\delta_r} \delta_r(t) + N_{\delta_t} \delta_t(t)$$

It has been shown that the yaw dynamics are augmented through the combined yaw rate of the gyro and the tail actuator system (Mettler, 2003). The inclusion of the tail gyro dynamics requires modelling the complex interaction of the engine drive train with the tail rotor. These parameters can only be determined using ground experiment and cannot be obtained during flight (Gavrilets et al., 2003) and therefore have been excluded in the estimation model.

The longitudinal and lateral blade flapping dynamics are described respectively by two coupled first-order differential equations.

$$\dot{a_1}(t) = -\frac{a_1(t)}{\tau_e} - q + A_{\delta_{lat}} \delta_{lat}(t) + A_{\delta_{lon}} \delta_{lon}(t)$$

$$\dot{b_1}(t) = -\frac{b_1(t)}{\tau_e} - p + B_{\delta_{lat}} \delta_{lat}(t) + B_{\delta_{lon}} \delta_{lon}(t)$$

The rotor time constant τ_e includes the influence of the stabiliser bar (Mettler et al., 1999). During online estimation these longitudinal and lateral time constants are represented as A_{a_1} and B_{b_1} respectively. The parametrized state space matrices can be defined as follows:

$$A = \begin{bmatrix} X_u & 0 & 0 & 0 & -g & 0 & 0 & 0 & 0 & X_{a_1} & X_{b_1} \\ 0 & Y_v & 0 & g & 0 & 0 & 0 & 0 & 0 & Y_{a_1} & Y_{b_1} \\ 0 & 0 & Z_w & 0 & 0 & 0 & 0 & 0 & 0 & Z_{a_1} & Z_{b_1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ L_u & L_v & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_{a_1} & L_{b_1} \\ M_u & M_v & 0 & 0 & 0 & 0 & 0 & 0 & 0 & M_{a_1} & M_{b_1} \\ 0 & N_v & 0 & 0 & 0 & 0 & 0 & 0 & N_r & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -\frac{1}{\tau_e} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -\frac{1}{\tau_e} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & Z_{\delta_{col}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & L_{\delta_{col}} & 0 & 0 \\ 0 & 0 & M_{\delta_{col}} & 0 & 0 \\ 0 & 0 & 0 & N_{\delta_r} & N_{\delta_t} \\ A_{\delta_{lon}} & 0 & 0 & 0 & 0 \\ 0 & B_{\delta_{lat}} & 0 & 0 & 0 \end{bmatrix}$$

Depending on the flight condition, some of the above-stated derivatives will be ignored as their effects will be negligible. The detailed description of the key derivatives is found in Section 4.1.5.

It needs to be noted that the yaw rate due to the engine torque N_{δ_t} can be accounted for only in certain helicopter models where the collective and throttle servos are not mixed. This is an available option in most RC transmitters.

4.5.7 Online estimation algorithm

RBF-based neural networks were used for the online estimation of the key derivatives described in Section 4.1.5. As discussed in Section 4.4.2, RBF networks have a major drawback in their inability to retain 'information' on previous input-output patterns. This would result in RBF learning at each time step making online model identification and parameter estimation impractical and memory intensive. To tackle this problem, an algorithm for the online estimation of aerodynamic parameters was developed. The algorithm flowchart is shown

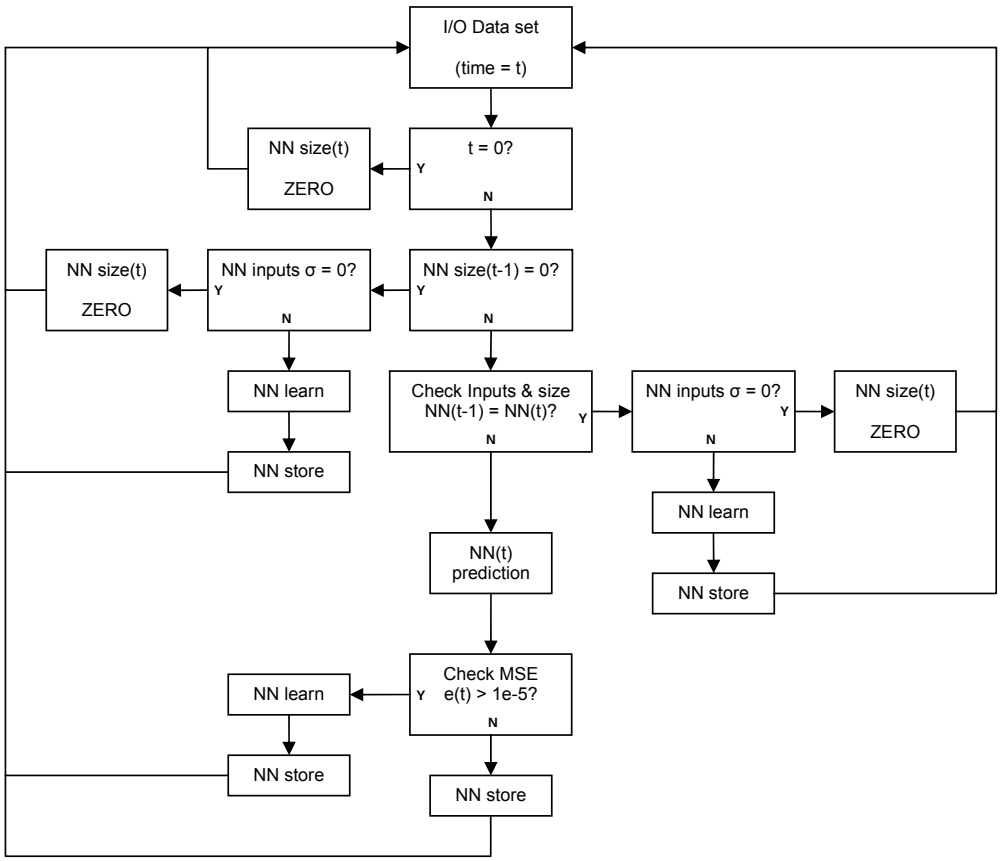

Figure 4.13: Online estimation algorithm flowchart.

in Figure 4.13. The moving window algorithm is used to generate an input-output data set at each time step (t). Once RBF learning of the data set is achieved, the network is stored. At the subsequent time step $(t+1)$, the new network input data is checked for steady state signals through computing their standard deviation σ and the redundant signals are eliminated. Prediction of the system underlying dynamics is validated by computing the mean square error (MSE). If the MSE is below the threshold value (1e-5), the network is stored and used in the time step $(t+2)$ (Singh, 2005).

This estimation algorithm has enabled the prediction capabilities of the RBF network to be applied in an online/real-time environment while minimizing computation requirements. Once the RBF network is stored, the DM and MDM methods are applied to compute the aerodynamic parameters as discussed in Section 4.5.3 and 4.5.4 respectively.

4.6 Results

The online model identification using neural networks was achieved using the RBF architecture. The translational accelerations a_x, a_y, a_z, the rotational accelerations $\dot{p}$, $\dot{q}$, $\dot{r}$ and the flapping rates $\dot{a}_1$, $\dot{b}_1$ were simulated using the following control inputs:

- $\delta_{lon} \longrightarrow a_x$, $\dot{q}$ and $\dot{a}_1$
- $\delta_{lat} \longrightarrow a_y$, $\dot{p}$ and $\dot{b}_1$
- $\delta_{col} \longrightarrow a_y$
- $\delta_r \longrightarrow \dot{r}$

The multistep 3211 flight maneuver was used for all control inputs during online identification and parameter estimation. As a measure to evaluate the network performance, the number of network neurons and memory usage was also recorded at each time step (0.02 sec). A moving window (stack) sizes of 50, 100 and 150 samples were used to generate input-output data sets. The ideal and turbulence cases are defined as follows:

- Ideal case: Calm wind, No sensor noise
- Turbulence case: Light turbulence of 3-5m/s and 5% sensor noise

Online model identification

The online identification model (see algorithm code in Appendix C on page 186) deleted input signals whose standard deviation and mean values were below values of 1e-6 and 1e-7 respectively. This was done to prevent matrix ill-conditioning to cause inversion errors during neuron weights computation. However, this approach introduced discontinuities as different stack patterns could potentially have different input space sizes which would result in a mismatch between the network input size and the stack input size. This problem was overcome by having a new network created each time the stack input size changed. This is illustrated on the flowchart in Figure 4.13. This caused the memory requirement to increase but was compensated by evaluating the mean square error (MSE) between the system's last recorded output and the network's predicted output. The network model, for a specific input-output space, was 're-created' only when the MSE rose above a threshold value of 1e-5. This approach acted as a network validation technique.

Online identification results are shown for hover flight on Figure 4.14 to Figure 4.37, forward 10m/s flight on Figure 4.38 to Figure 4.61 and forward 20m/s flight on Figure 4.62 to Figure 4.88. The effect of state and measurement noise was investigated through atmospheric turbulence and sensor noise resulting from the flight test instrumentation. The simulation facility had the following specifications:

- Processor: 1.73GHz Intel
- Memory: 504MB of RAM

The effect of atmospheric disturbance and sensor noise can be clearly seen in Figure 4.14. As shown in Section 4.4.2, this has an adverse effect on network size on Figure 4.15 and memory usage on Figure 4.16. The latter is given as a percentage of available memory. The higher the required memory for a particular stack size configuration, the higher the convergence time for online model identification.

The network is said to have saturated and unable to identify the input-output set when the network size equals the stack size used for identification. $\dot{r}$ identification for hover flight in Figure 4.30, forward 10m/s flight in Figure 4.54 and forward 20m/s flight in Figure 4.81, showed network saturation for all three stack sizes. The adverse effect of network saturation is shown in a_x identification in Figure 4.64 for the turbulence case. The network saturated with a stack of 150 resulting in a memory usage of 85% shown in Figure 4.64. The network saturation occurs when: (1) the signal-to-noise ratio decreases in the input-output data thereby preventing the network to efficiently select the significant neuron centers, (2) matrix ill-conditioning for network weights computation caused by input vector scaling.

The neural network model identification robustness to noise was investigated by including state and measurement noise. Good noise attenuation results was achieved with a_x identification for all flight conditions except the 20m/s forward flight with a network size between 6 to 8 neurons shown in Figure 4.15 and Figure 4.39 respectively. The $\dot{q}$ identification shows similar results with a network size between 10 to 12 neurons shown in Figure 4.27 and Figure 4.51 respectively. The $\dot{r}$ hover flight identification shows network instability in the turbulence case in Figure 4.30. This could be caused by insufficient signal content in the network inputs for that time period. The same effect can be seen for the a_z forward 10m/s flight identification in Figure 4.45. The smaller stack size of 50 samples was unaffected as the smaller moving window allows the network to have a higher forgetting factor. The identification of the rotor flapping dynamics $\dot{a}_1$ and $\dot{b}_1$ was achieved for all flight conditions in the presence of state and measurement noise.

Online parameter estimation

The online estimation of the stability and control derivatives using RBFN-based DM and MDM methods was achieved for all flight conditions namely: hover, forward 10m/s flight and forward 20m/s flight. This is shown in Figure 4.89 to Figure 4.198 where all the estimated parameters used a stack size of 150. 95% confidence intervals were calculated at each time step for both the ideal and turbulence cases. The estimated parameters were then compared with the linearized model parameters. It was assumed that the reference linear model parameters stay constant throughout the flight condition. For each estimated parameter, the stack size was varied (between 50 and 150 samples) and the number of estimated val-

ues within 95%(high confidence) and 60%(low confidence) of the reference value are stored and shown in Table 4.2 to Table 4.56. The estimation results are classified with respect to the known helicopter dynamics namely: roll and pitch dynamics, longitudinal and lateral dynamics, heave dynamics, yaw dynamics and flapping/stabiliser bar dynamics.

The longitudinal derivatives X_u, X_θ, X_{a_1} and the lateral derivatives Y_v, Y_ϕ, Y_{b_1}, were identified and compared with the equivalent parameter from the linear models. The derivative X_u could not be identified using the MDM in turbulence for hover flight in Figure 4.90, forward 10m/s flight in Figure 4.126 and forward 20m/s flight in Figure 4.162 with a maximum of 2 values of 95% confidence shown in Table 4.38. This could be due to the decrease in the signal-to-noise ratio in the network inputs. In the MDM ideal case, the number of high confidence values decrease from hover to high speed forward flight from 265 to 0. The DM method produces the opposite with the number of high confidence values increasing from 0 to 47. The derivative Y_v shows consistently larger deviations with the DM method than the MDM method for the ideal cases. This can be clearly seen for hover flight in Figure 4.96, forward 10m/s flight in Figure 4.132 and forward 20m/s flight in Figure 4.168. This is expected from the DM method as absolute values are computed as compared to differential values for the MDM method although the MDM method is susceptible to large outliers in areas of steady state variables. Similar results were obtained for the inertial derivative X_θ with the forward 10m/s flight producing the lowest number of estimated values for both the DM and MDM methods in turbulence shown in Figure 4.127 and Figure 4.128 respectively. The opposite resulted with Y_ϕ. The forward 10m/s flight MDM method produced 387 values with 95% confidence in turbulence as compared to the 18 for the DM method shown in Table 4.24. The rotor force derivatives X_{a_1} and Y_{b_1} shows poor results in all flight conditions for the DM method in turbulence with the worst estimation at forward 10m/s flight shown in Table 4.22 and Table 4.25 respectively. This could be due to the increased sensitivity to control inputs resulting in higher deviations.

The roll derivatives L_v,L_{b_1} and pitch derivatives M_{a_1},$M_{\delta_{col}}$ are identified. The speed derivative L_v shows poor results in all flight conditions for the DM method in turbulence with the best estimation at forward 10m/s flight shown in Table 4.22. This could be the result of poor signal content and high sensitivity to control inputs resulting in large deviation as shown in Figure 4.141. The MDM time simulations can be seen for hover in Figure 4.108, forward 10m/s flight in Figure 4.144 and forward 20m/s flight in Figure 4.180. The rotor moment derivative L_{b_1} shows an increase in high confidence MDM-based estimated values from 116 in hover flight to 333 in forward 20m/s flight in turbulence. In the ideal case, DM-based estimated values varied from 294 in hover flight to 410 in forward 20m/s flight. This is shown in Table 4.11 and Table 4.47 respectively. This could be due to the forward speed increasing the sensitivity of flapping angles to control inputs, resulting in the MDM method having better robustness qualities in the presence of state and measurement noise. M_{a_1} shows a similar trend with an increase of MDM-based estimated values from 48 in hover flight to 443 in forward 10m/s flight in turbulence. This is shown in Table 4.12 and table 4.30 respectively. No DM-based estimated values could be identified in the forward 20m/s flight in turbulence as shown in Figure 4.181. $M_{\delta_{col}}$ could only identified in forward 20m/s flight due to the

high signal content and high sensitivity to control inputs. The ideal case shows good results with 351 high confidence MDM-based values using a stack of 150 shown in Figure 4.184. No DM-based estimated values could be identified in turbulence as shown in Figure 4.183.

The heave derivatives Z_w and $Z_{\delta_{col}}$ are identified. The damping derivative Z_w shows poor results in all flight conditions for the DM method in turbulence with the worst estimation at forward 20m/s flight shown in Table 4.44. The MDM-based low confidence estimated values increased from 0 in hover flight to 145 in forward 20m/s flight in turbulence. This is shown in Figure 4.102 and Figure 4.174 respectively. This is expected as the identification results showed network saturation for hover flight in turbulence in Figure 4.20 and network instability for forward 10m/s flight in Figure 4.45. The collective derivative $Z_{\delta_{col}}$ showed a similar trend with the MDM-based low confidence estimated values increasing from 0 in hover flight to 296 in forward 20m/s flight in turbulence. This is shown in Figure 4.104 and Figure 4.176 respectively. In the ideal case, DM-based high confidence estimated values increase from 351 in hover flight to 385 in forward 20m/s flight. This is shown in Table 4.9 and Table 4.45 respectively. This could be caused by a higher signal content as the helicopter moves from hover flight to forward cruise flight.

The yaw derivatives N_r, N_{δ_r} and N_{δ_t} were identified. The damping derivative N_r had poor results in all flight conditions. This could be due to the active yaw damping system producing a negative feedback to the yaw dynamics making open-loop identification and parameter estimation difficult. Poor identification results through network saturation and network instability in hover flight in Figure 4.30, forward 10m/s flight in Figure 4.54 and forward 20m/s flight 20m/s in Figure 4.81. The derivative N_{δ_r} shows the MDM-based low confidence estimated values increasing from 0 in hover flight to 152 in forward 20m/s flight in turbulence. This is shown in Table 4.14 and Table 4.51 respectively. The derivative N_{δ_t} shows a similar trend with the MDM-based low confidence estimated values increasing from 36 in hover flight to 219 in forward 20m/s flight in turbulence. This is shown in Table 4.14 and Table 4.51 respectively. At stated earlier, these poor results could be the result of poor signal content and the presence of the yaw damping control system.

The flapping/stabiliser bar derivatives A_{a_1}, $A_{\delta_{lon}}$ and B_{b_1}, $B_{\delta_{lat}}$ were identified. The A_{a_1} and B_{b_1} represent the inverse of rotor time constant τ_e with the influence of the stabiliser bar. The derivative A_{a_1} shows the MDM-based high confidence estimated values increasing from 16 in hover flight to 351 in forward 20m/s flight in turbulence. This is shown in Table 4.16 and Table 4.53 respectively. In the ideal case, the DM-based high confidence estimated values decrease from 359 in hover flight to 130 in forward 20m/s flight in turbulence. This could be the result of the DM method inability to predict the underlying model dynamics and deteriorating effect of the increased sensitivity to control inputs. The derivative $A_{\delta_{lon}}$ also shows consistent results with MDM-based high confidence estimated values increasing from 16 in hover flight to 351 in forward 20m/s flight in turbulence. This is shown in Table 4.17 and Table 4.54 respectively. The derivative B_{b_1} shows poor results in all flight conditions in the presence of turbulence. This could be attributed to low signal-to-noise ratio and high signal content requirements. The derivative $B_{\delta_{lat}}$ shows the MDM-based high confidence estimated

values increasing from 158 in hover flight to 351 in forward 20m/s flight in turbulence. This is shown in Table 4.19 and Table 4.56 respectively.

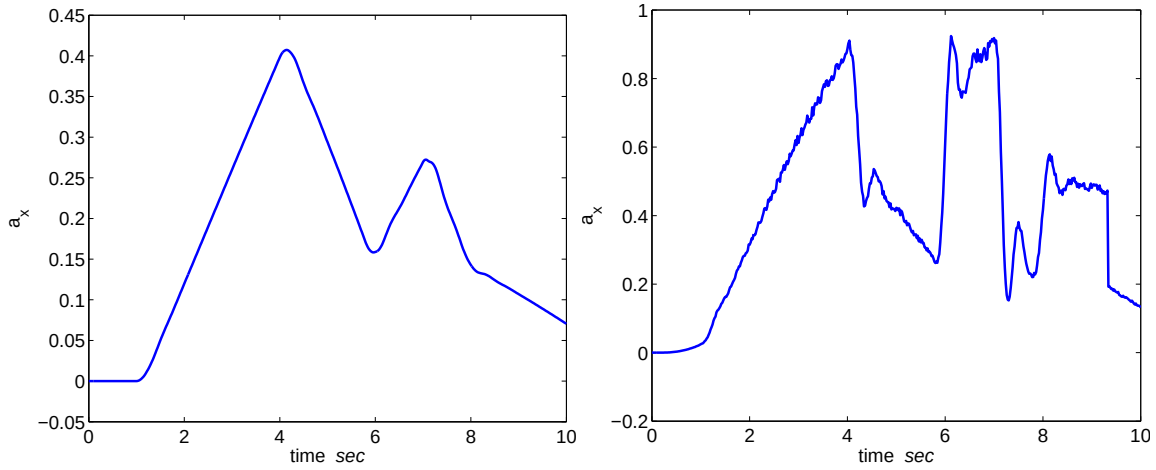

Figure 4.14: a_x hover flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

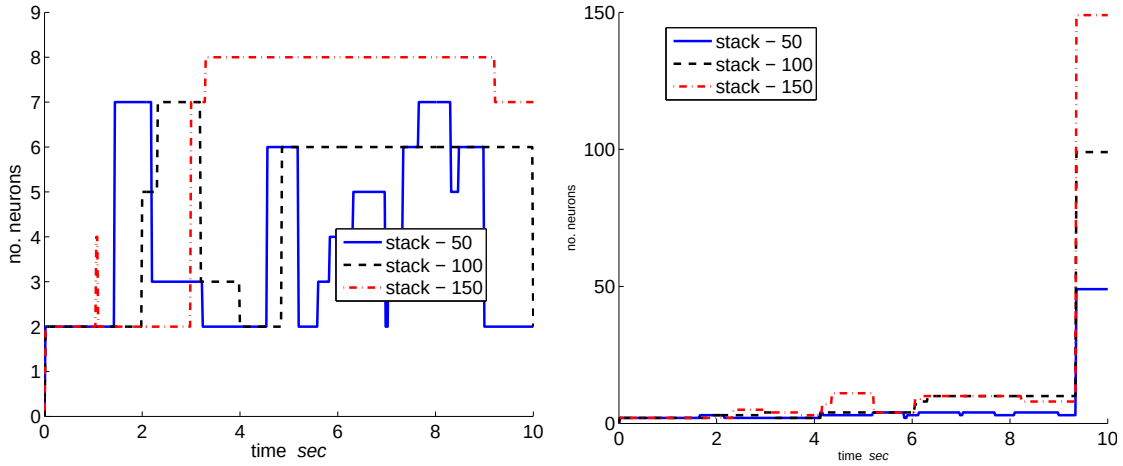

Figure 4.15: a_x hover flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

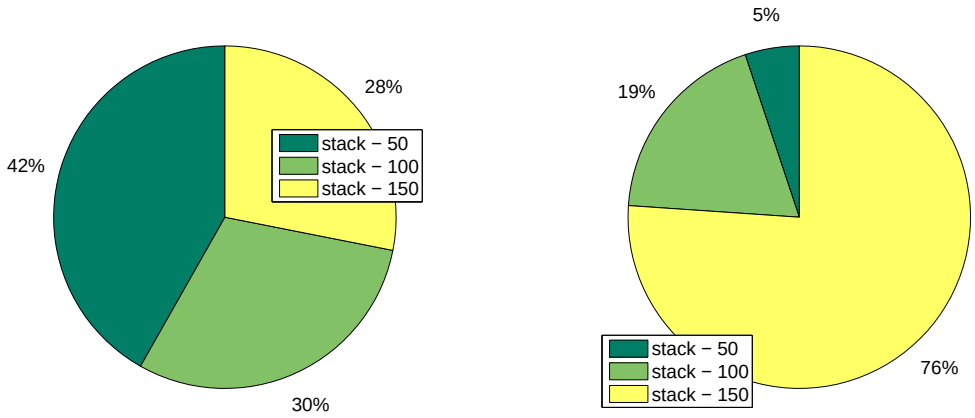

Figure 4.16: a_x hover flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

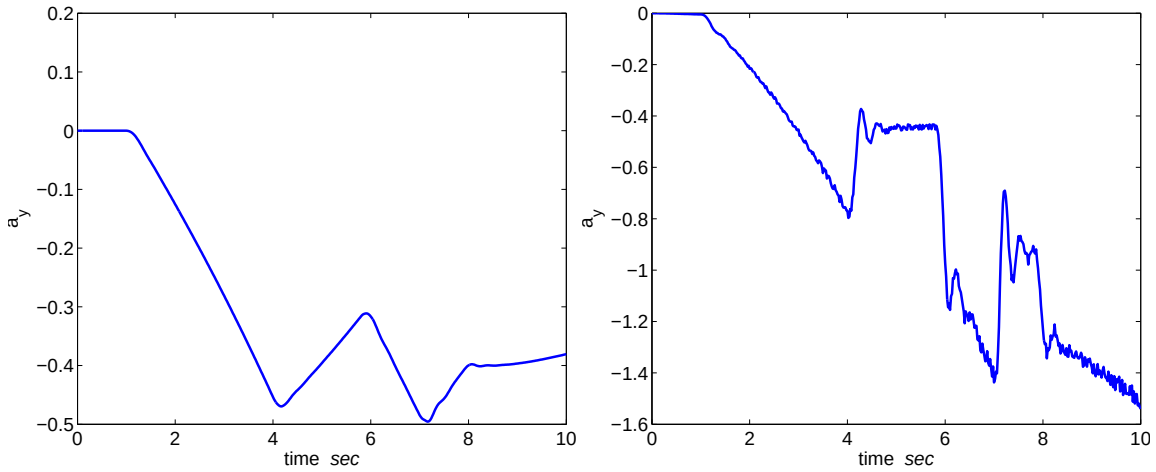

Figure 4.17: a_y hover flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

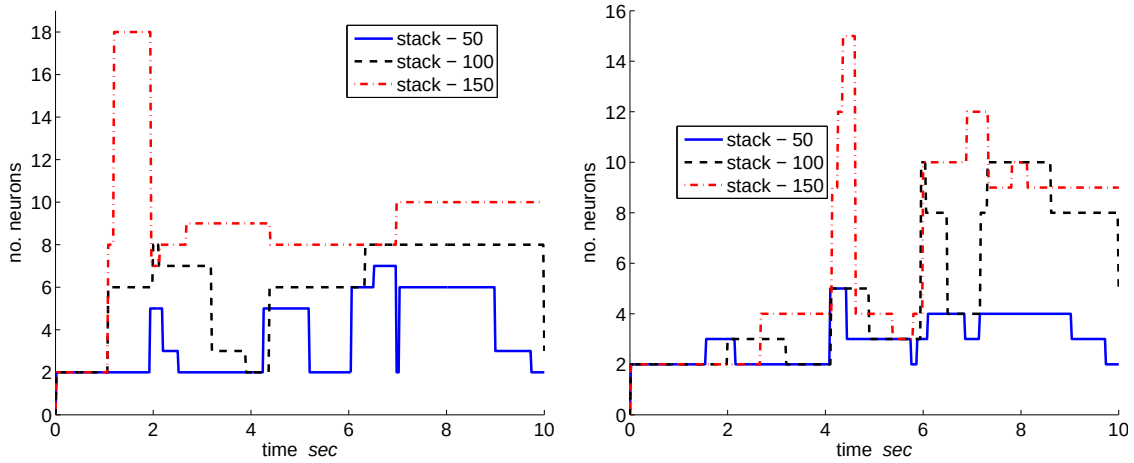

Figure 4.18: a_y hover flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

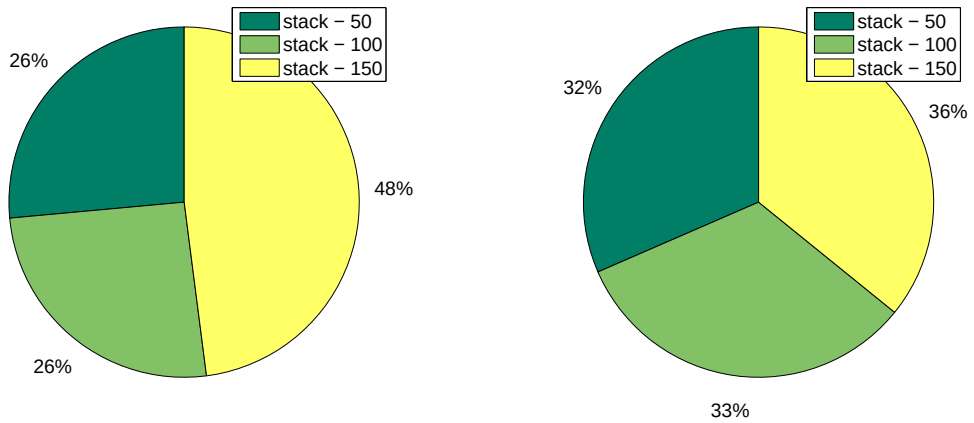

Figure 4.19: a_y hover flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

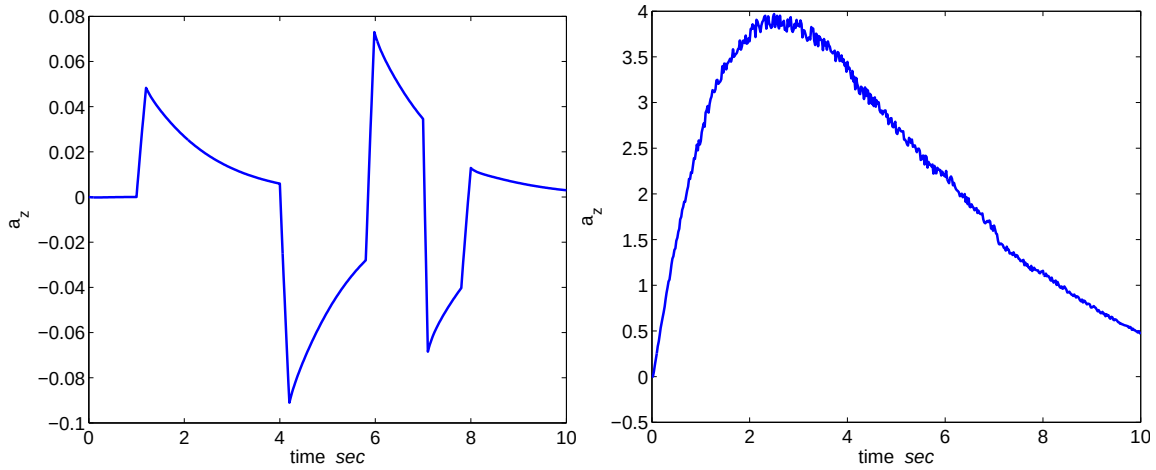

Figure 4.20: a_z hover flight online identification using δ_{col}: Time history (1) ideal, (2) turbulence.

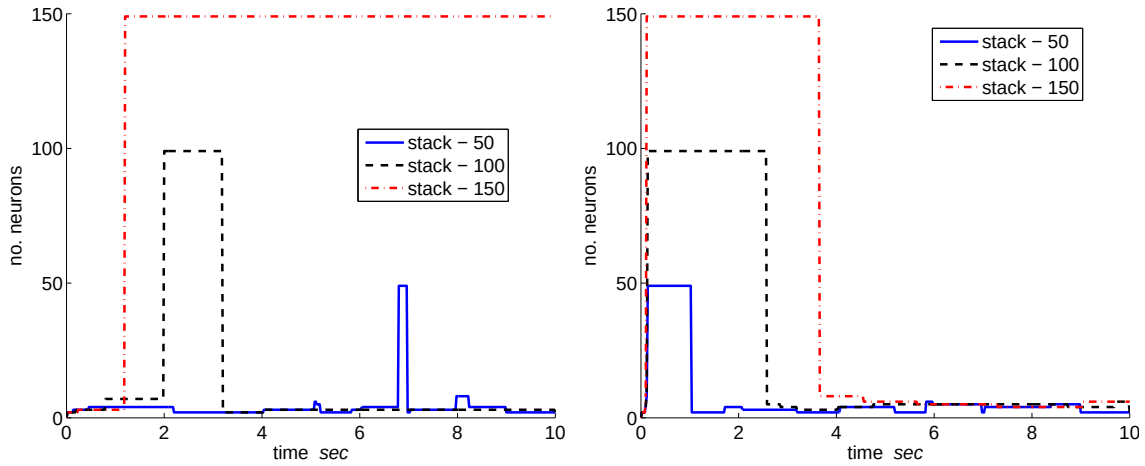

Figure 4.21: a_z hover flight online identification using δ_{col}: Network neurons (1) ideal, (2) turbulence.

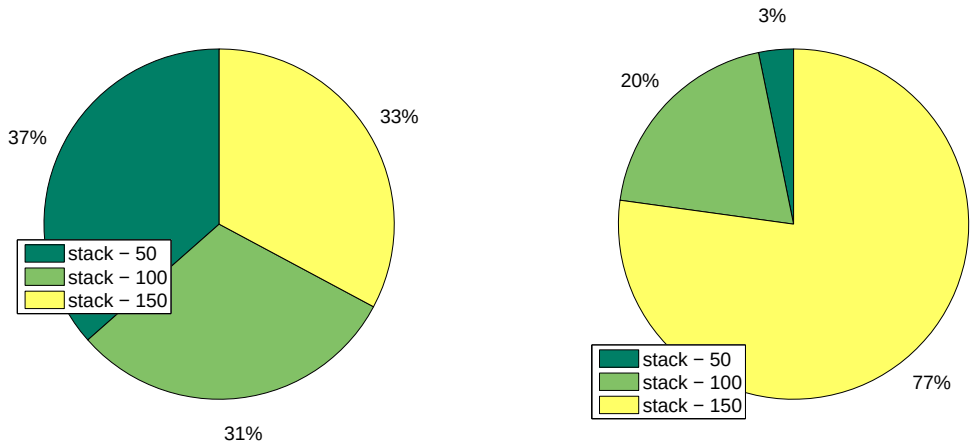

Figure 4.22: a_z hover flight online identification using δ_{col}: Memory usage (1) ideal, (2) turbulence.

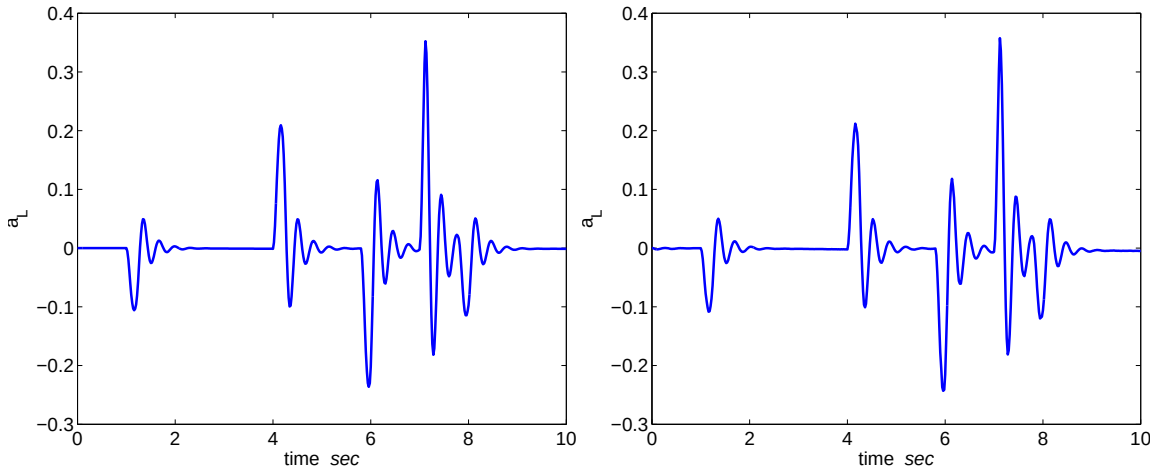

Figure 4.23: $\dot{p}$ hover flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

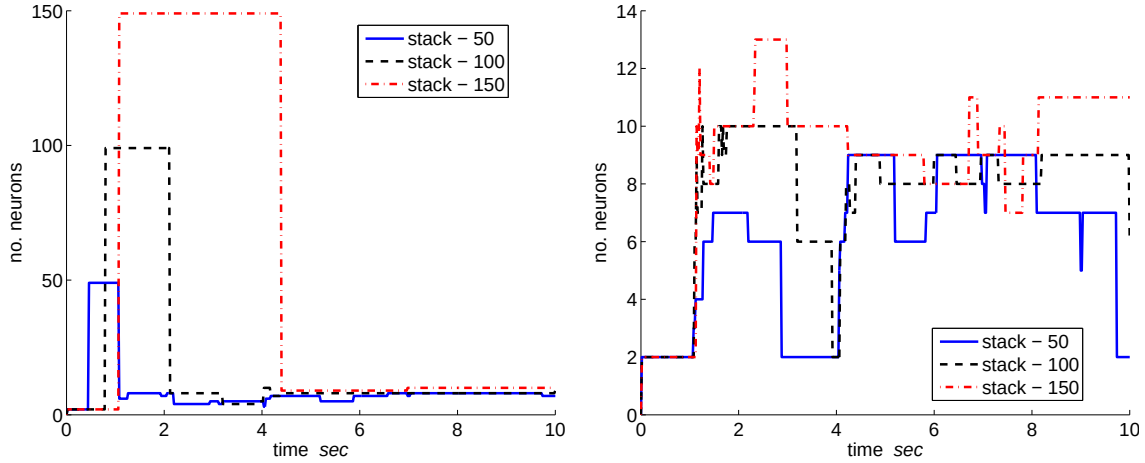

Figure 4.24: $\dot{p}$ hover flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

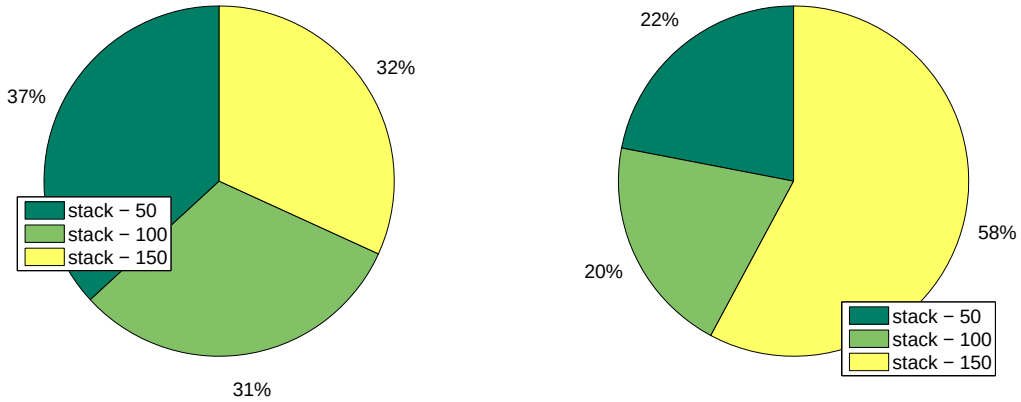

Figure 4.25: $\dot{p}$ hover flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

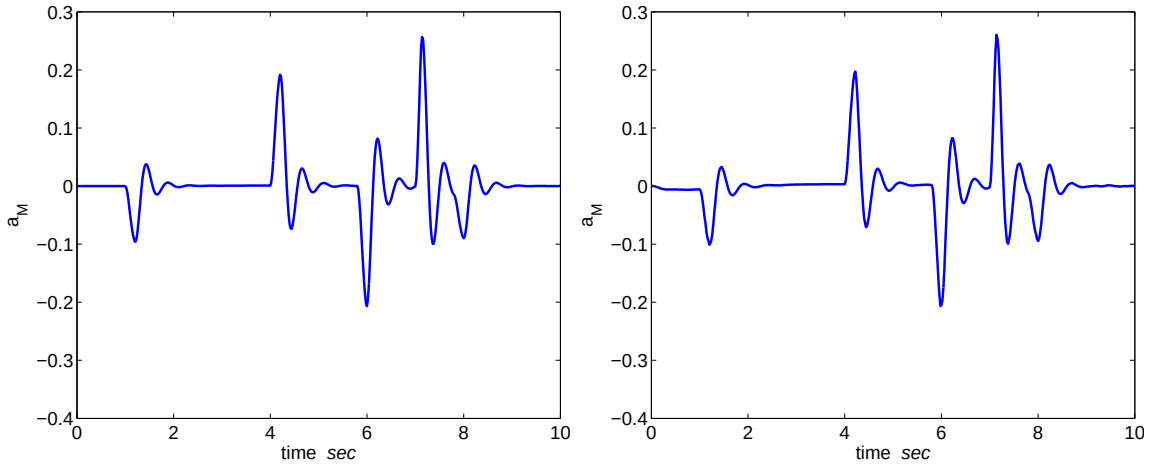

Figure 4.26: $\dot{q}$ hover flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

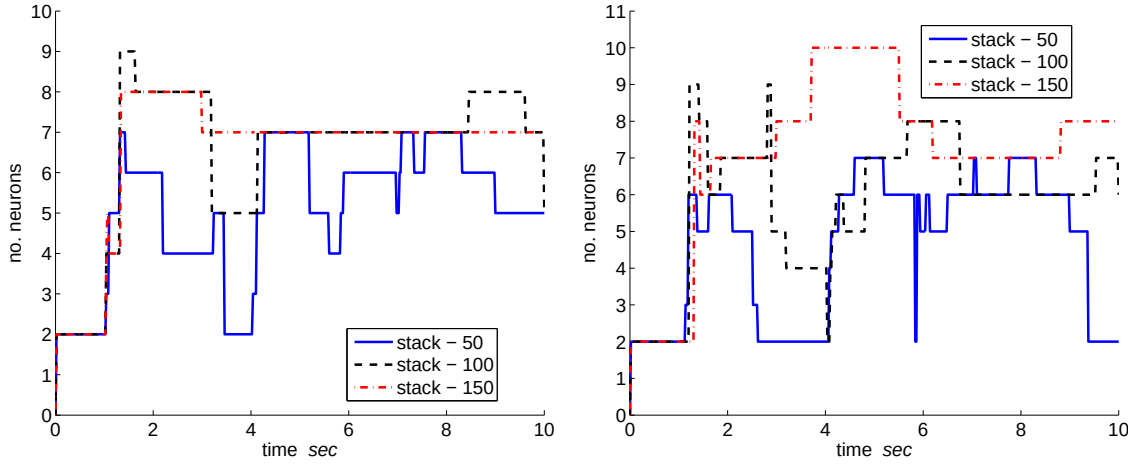

Figure 4.27: $\dot{q}$ hover flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

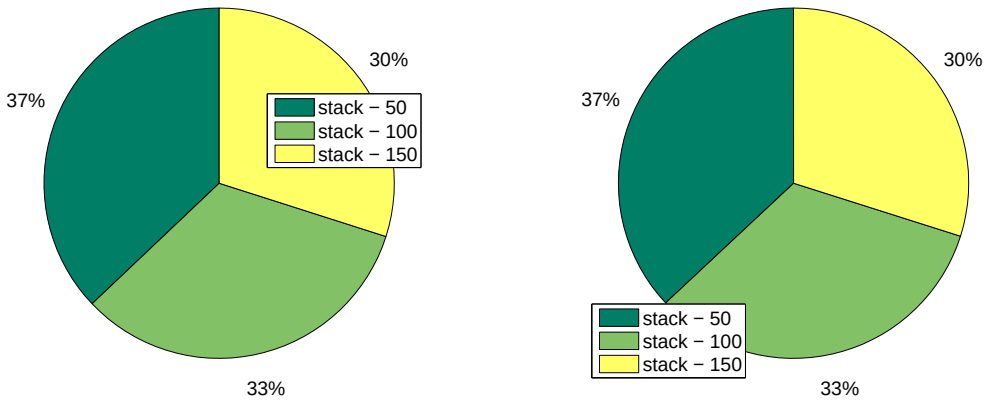

Figure 4.28: $\dot{q}$ hover flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

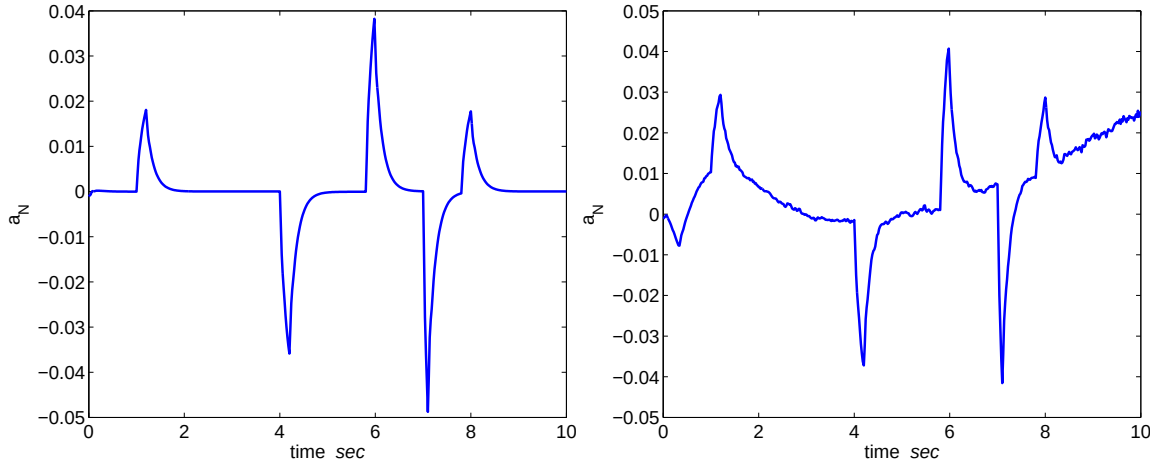

Figure 4.29: $\dot{r}$ hover flight online identification using δ_r: Time history (1) ideal, (2) turbulence.

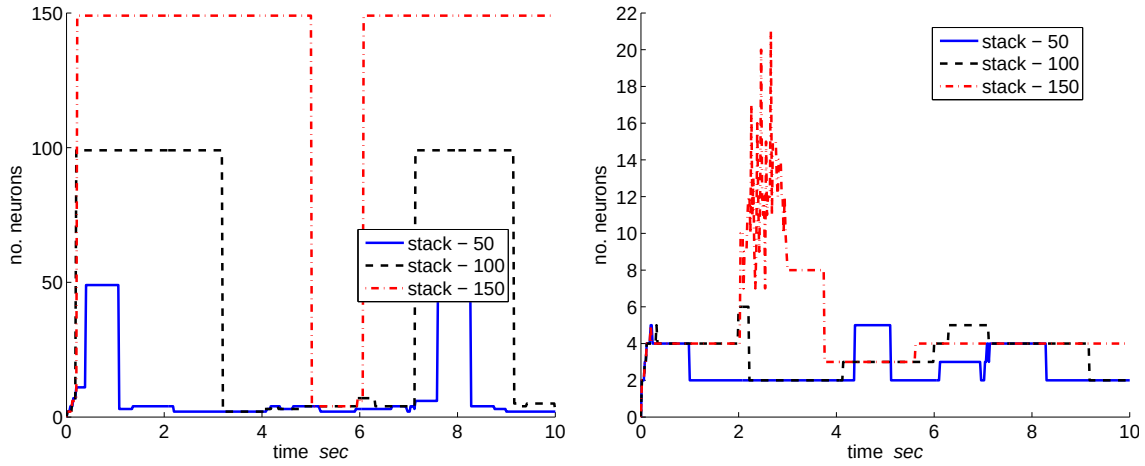

Figure 4.30: $\dot{r}$ hover flight online identification using δ_r: Network neurons (1) ideal, (2) turbulence.

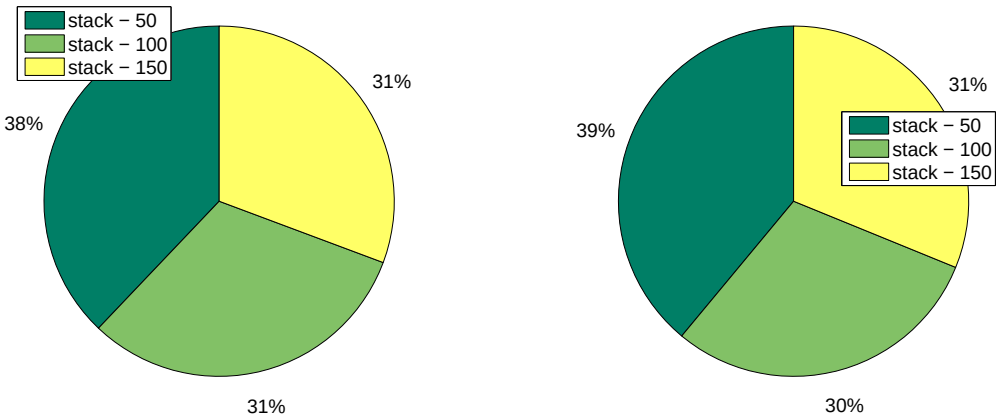

Figure 4.31: $\dot{r}$ hover flight online identification using δ_r: Memory usage (1) ideal, (2) turbulence.

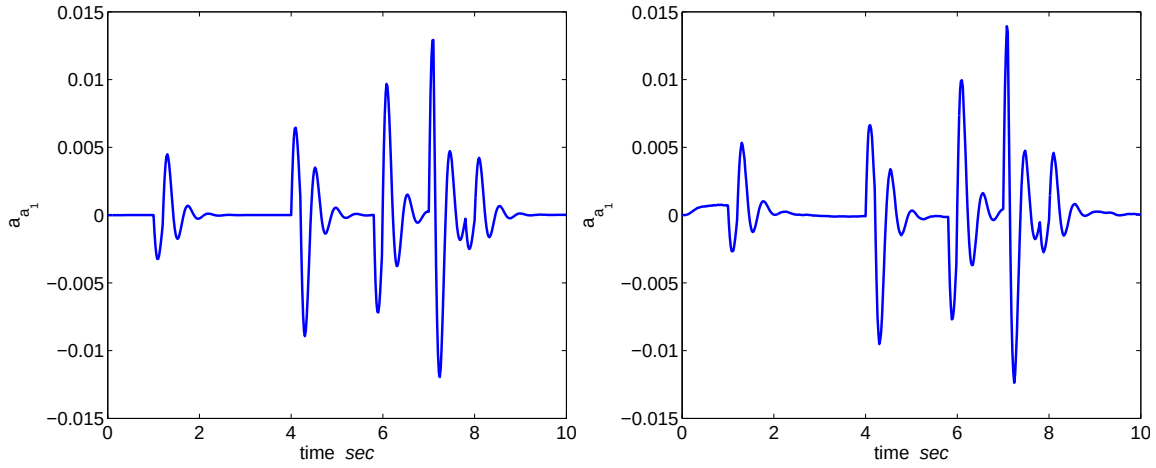

Figure 4.32: $\dot{a}_1$ hover flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

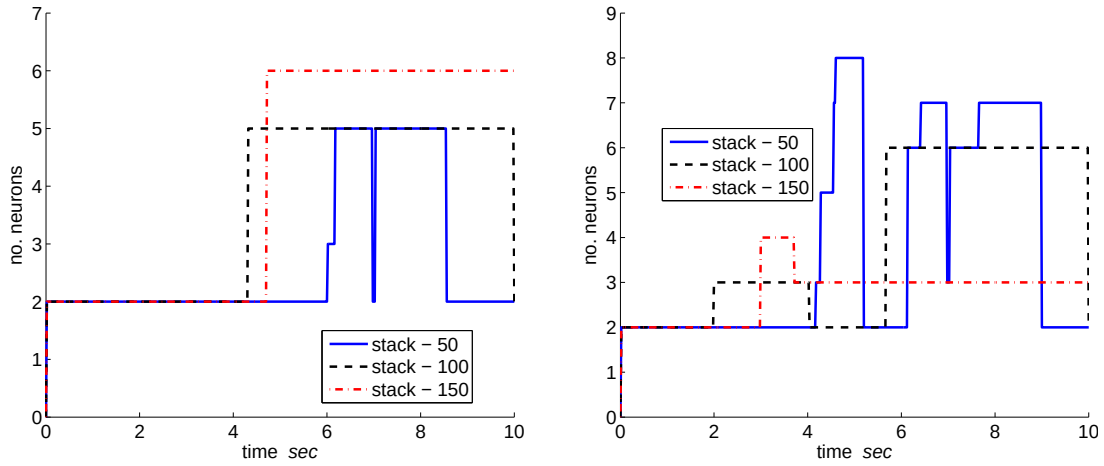

Figure 4.33: $\dot{a}_1$ hover flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

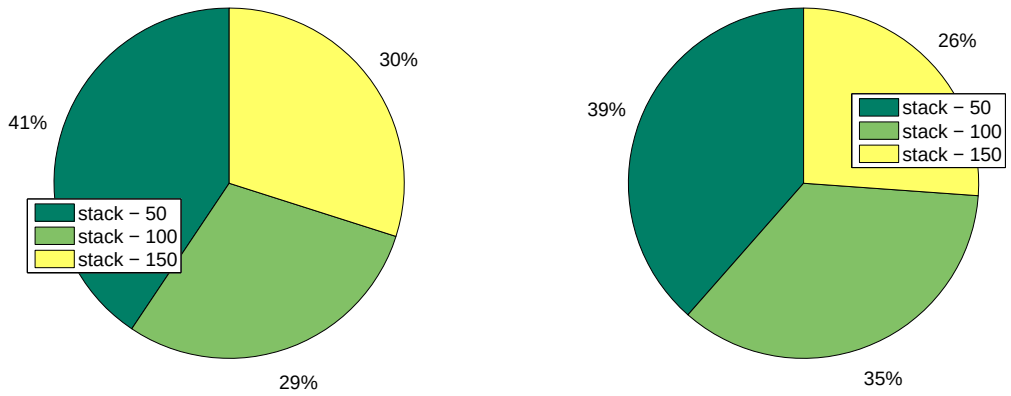

Figure 4.34: $\dot{a}_1$ hover flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

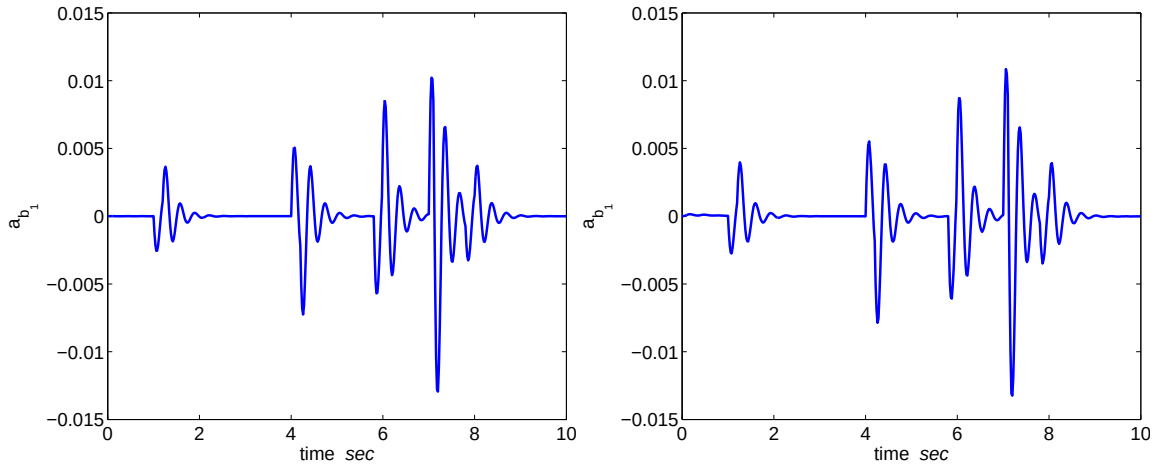

Figure 4.35: $\dot{b}_1$ hover flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

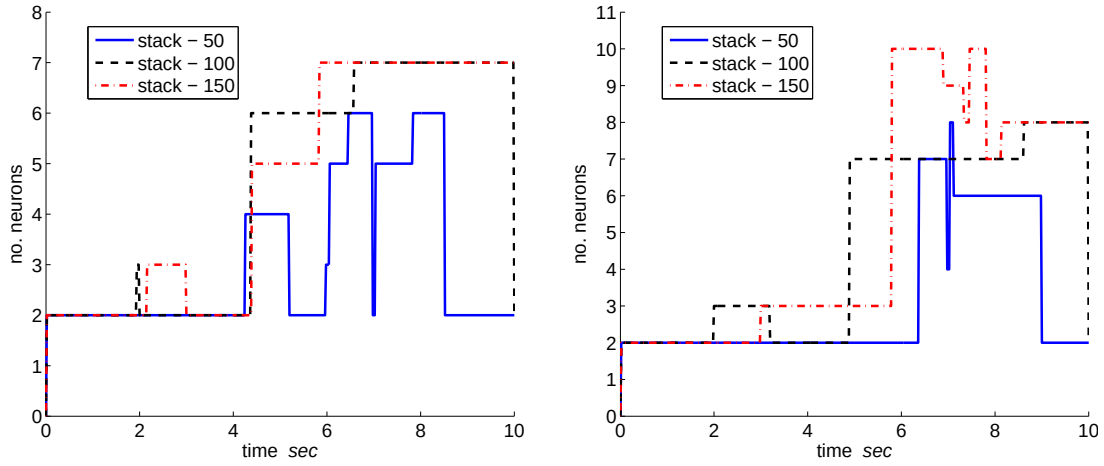

Figure 4.36: $\dot{b}_1$ hover flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

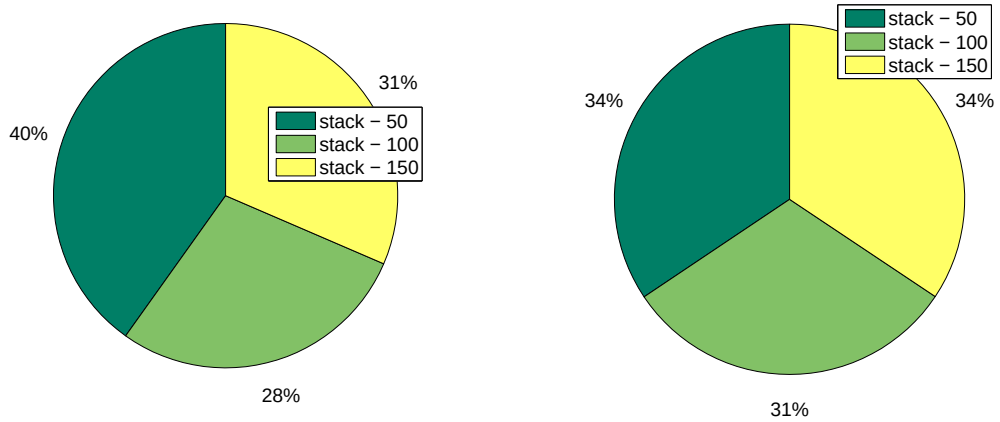

Figure 4.37: $\dot{b}_1$ hover flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

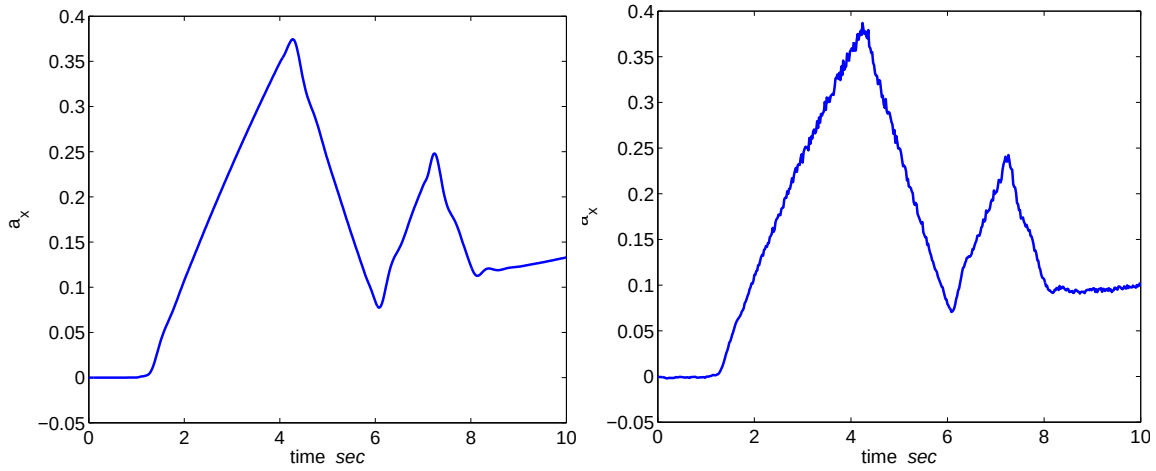

Figure 4.38: a_x forward 10m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

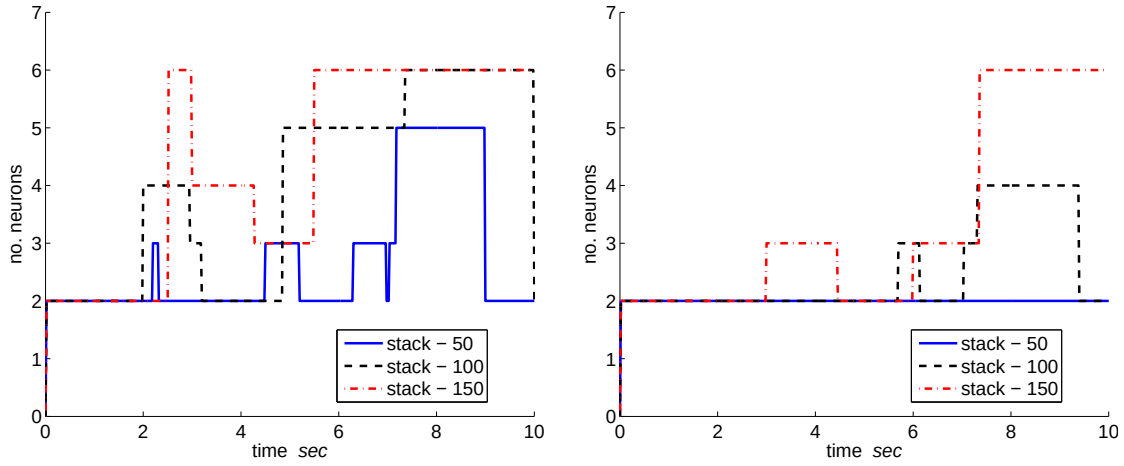

Figure 4.39: a_x forward 10m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

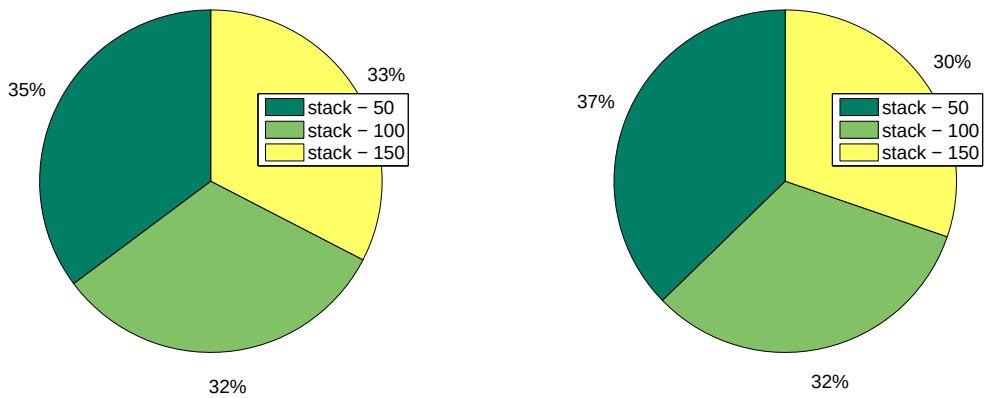

Figure 4.40: a_x forward 10m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

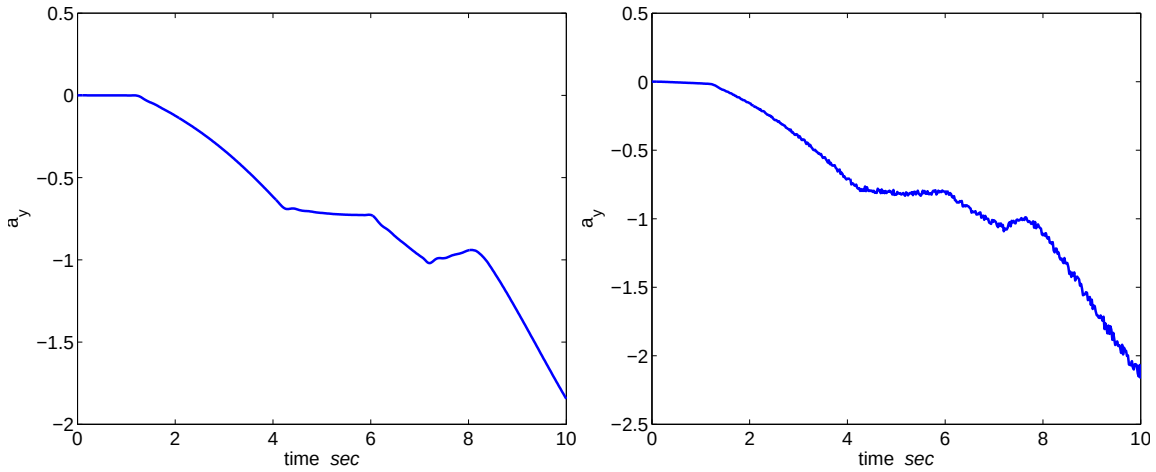

Figure 4.41: a_y forward 10m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

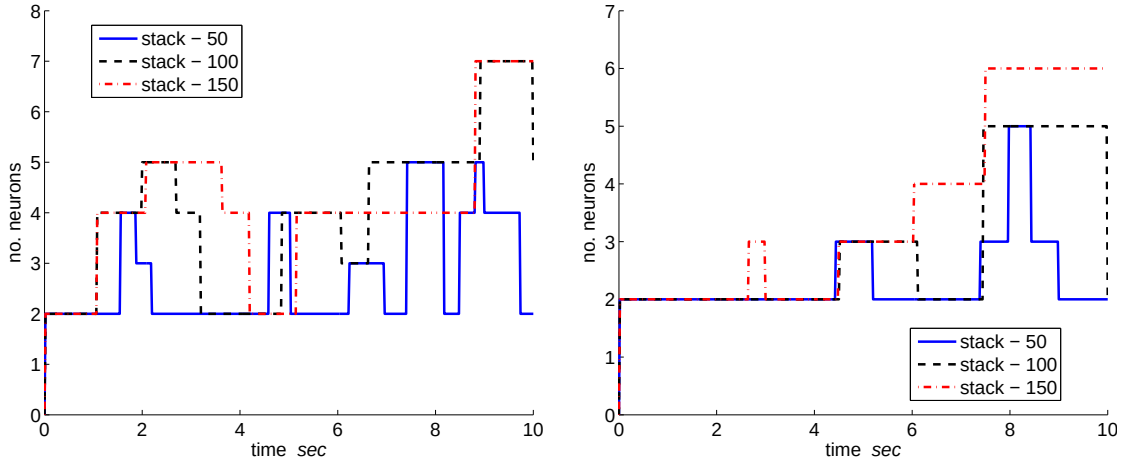

Figure 4.42: a_y forward 10m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

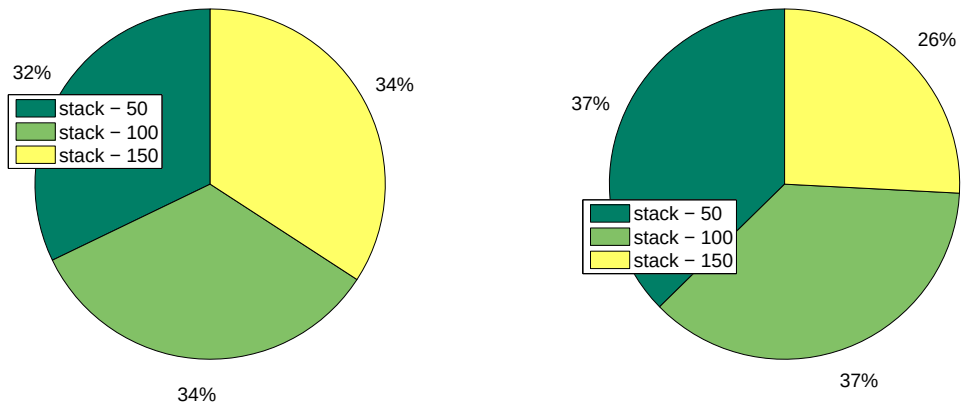

Figure 4.43: a_y forward 10m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

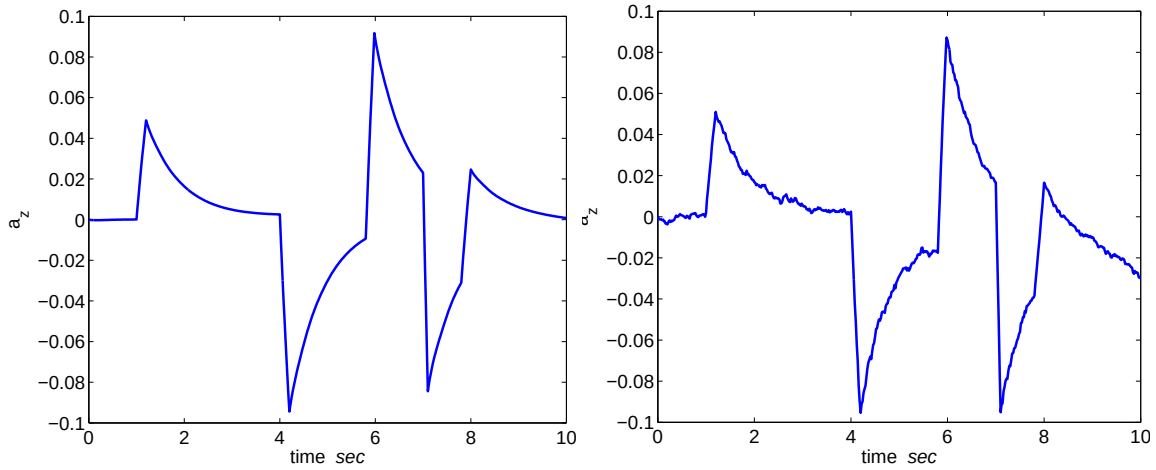

Figure 4.44: a_z forward 10m/s flight online identification using δ_{col}: Time history (1) ideal, (2) turbulence.

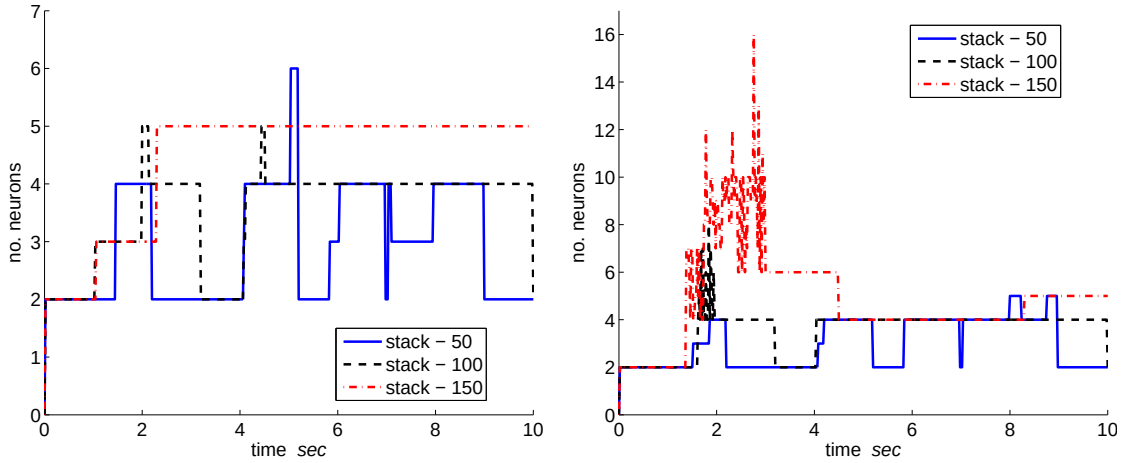

Figure 4.45: a_z forward 10m/s flight online identification using δ_{col}: Network neurons (1) ideal, (2) turbulence.

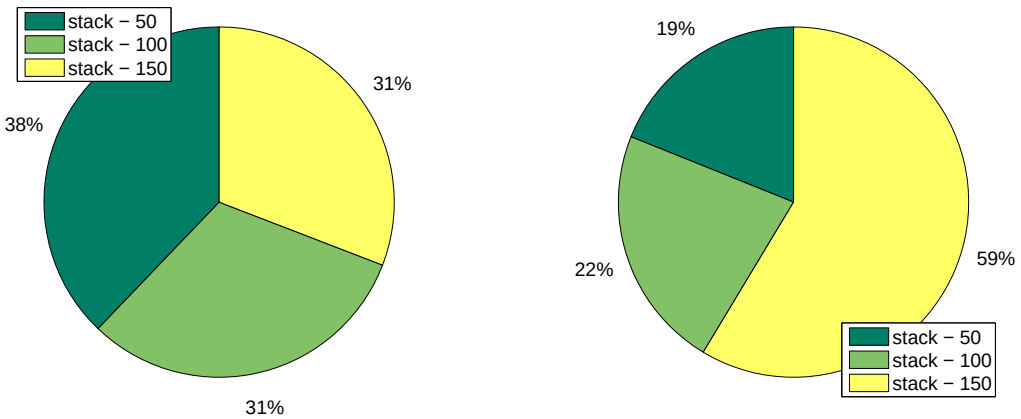

Figure 4.46: a_z forward 10m/s flight online identification using δ_{col}: Memory usage (1) ideal, (2) turbulence.

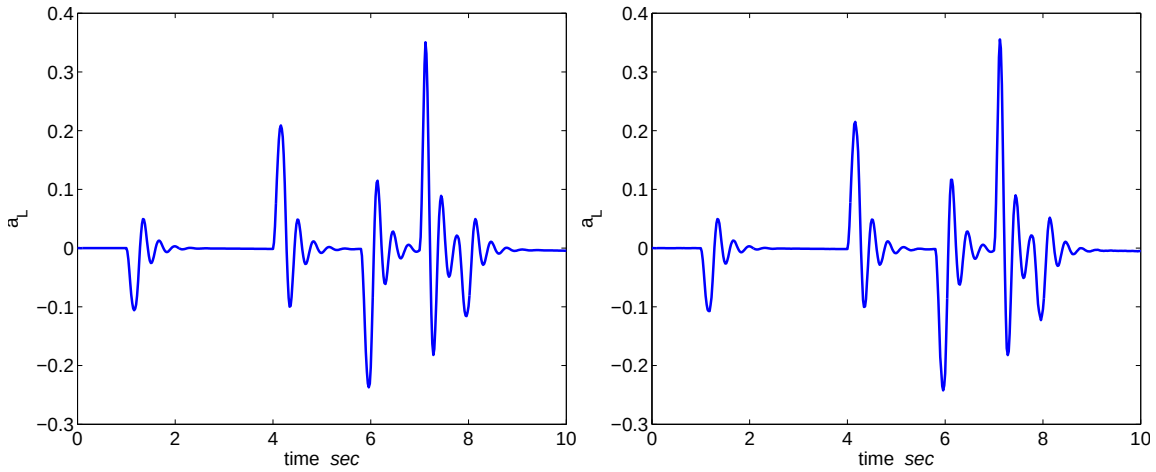

Figure 4.47: $\dot{p}$ forward 10m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

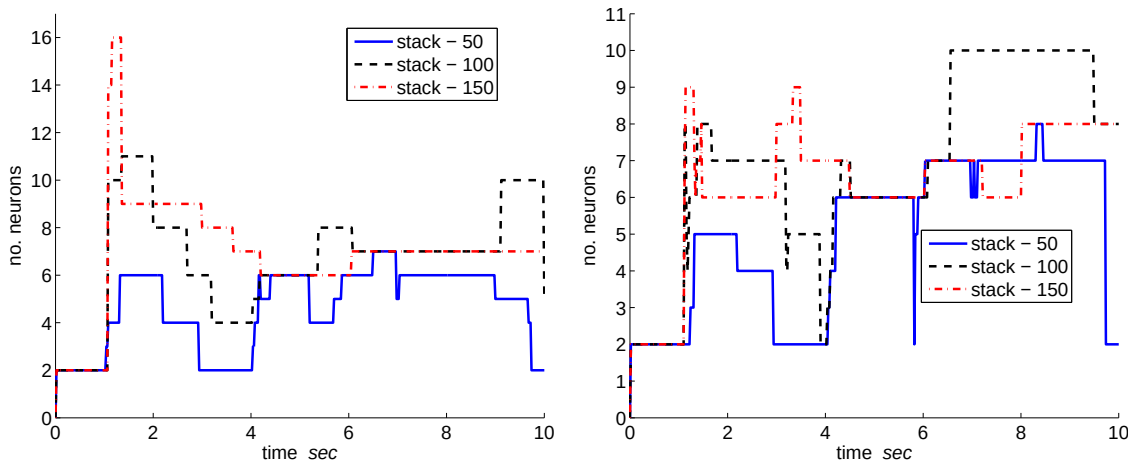

Figure 4.48: $\dot{p}$ forward 10m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

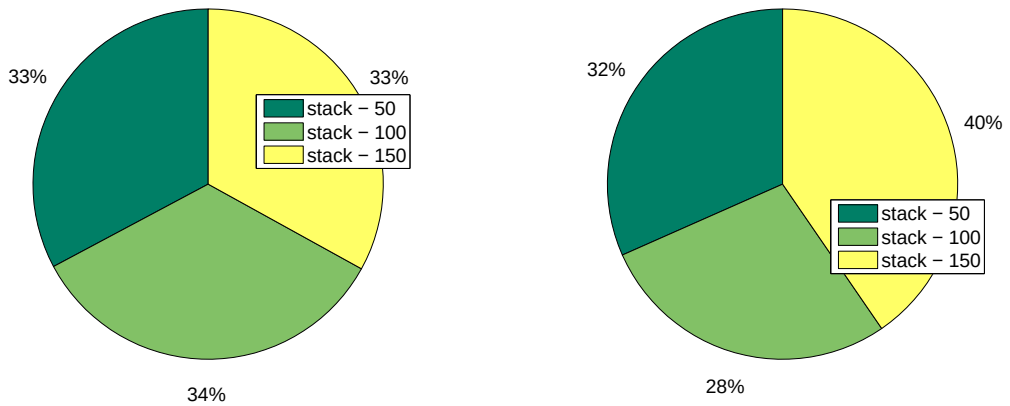

Figure 4.49: $\dot{p}$ forward 10m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

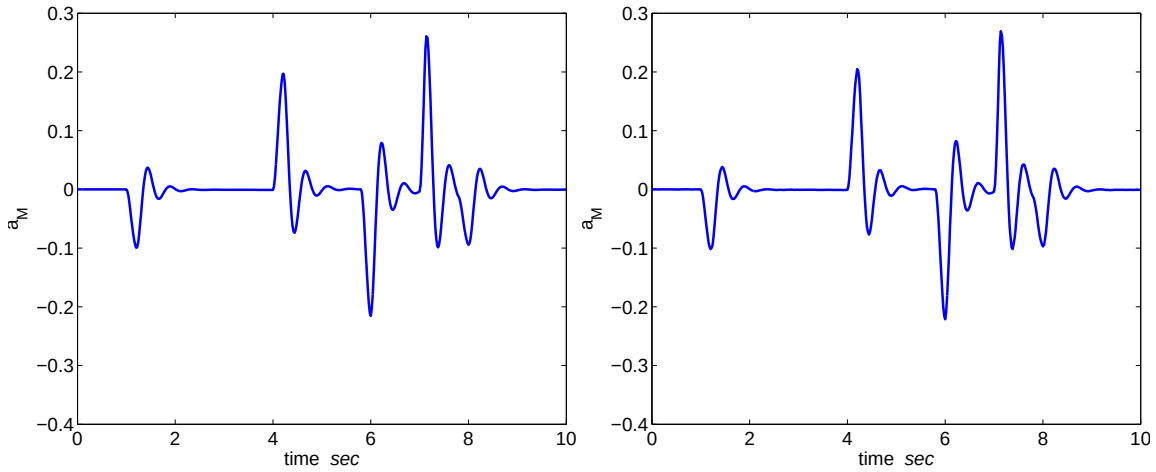

Figure 4.50: $\dot{q}$ forward 10m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

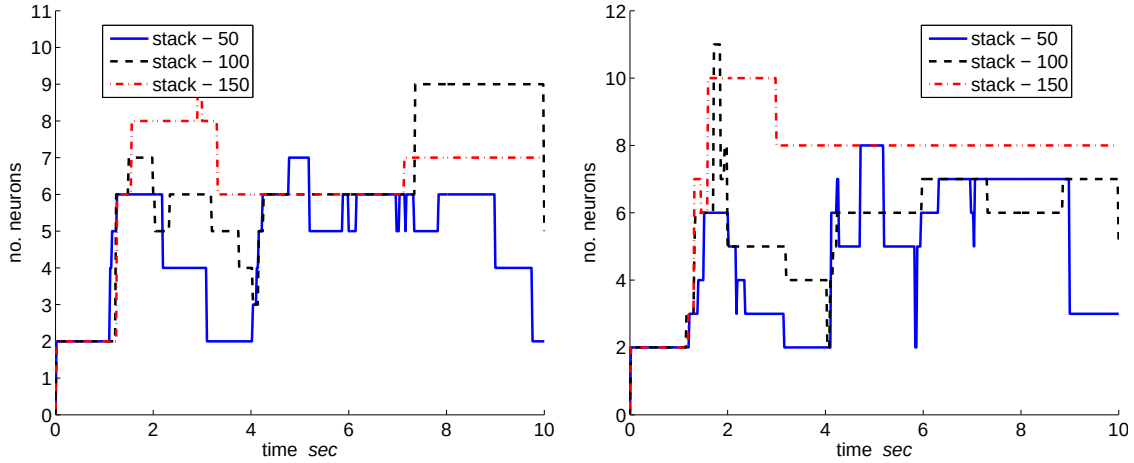

Figure 4.51: $\dot{q}$ forward 10m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

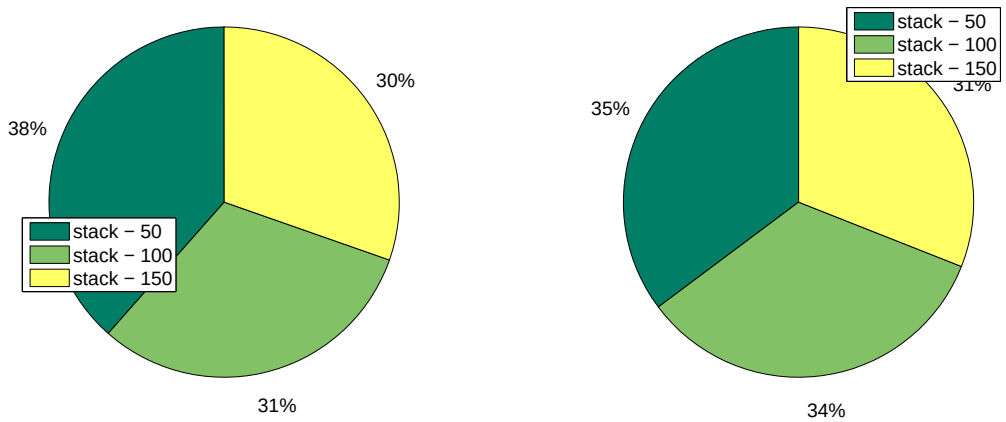

Figure 4.52: $\dot{q}$ forward 10m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

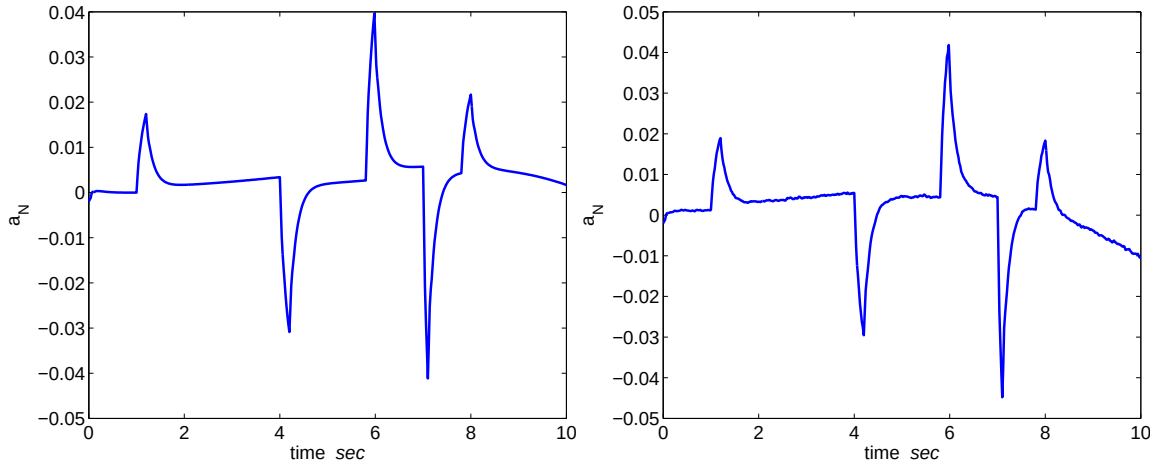

Figure 4.53: $\dot{r}$ forward 10m/s flight online identification using δ_r: Time history (1) ideal, (2) turbulence.

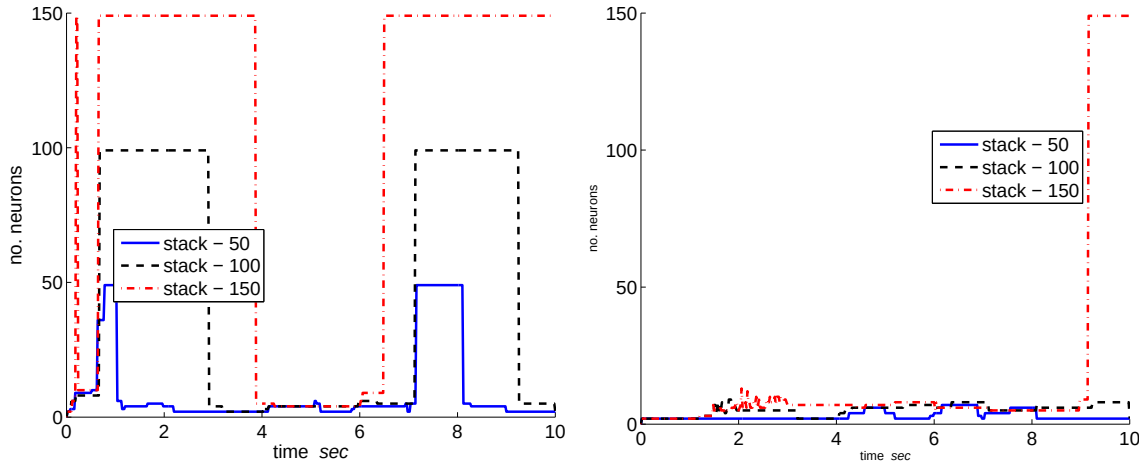

Figure 4.54: $\dot{r}$ forward 10m/s flight online identification using δ_r: Network neurons (1) ideal, (2) turbulence.

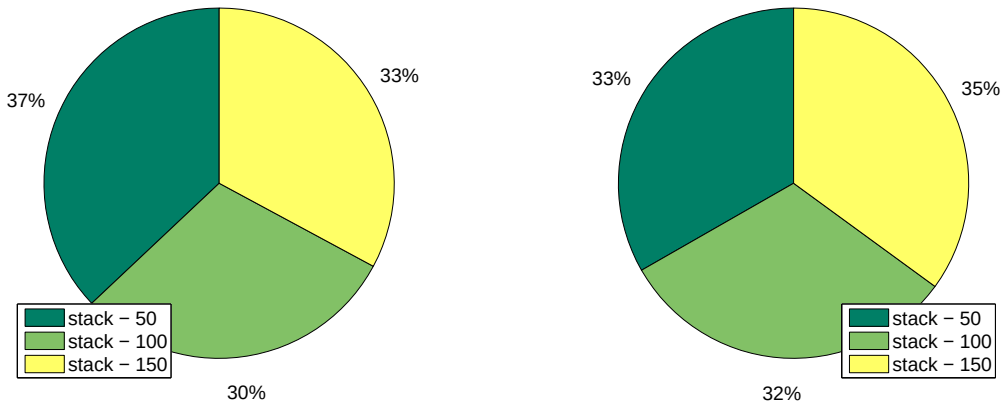

Figure 4.55: $\dot{r}$ forward 10m/s flight online identification using δ_r: Memory usage (1) ideal, (2) turbulence.

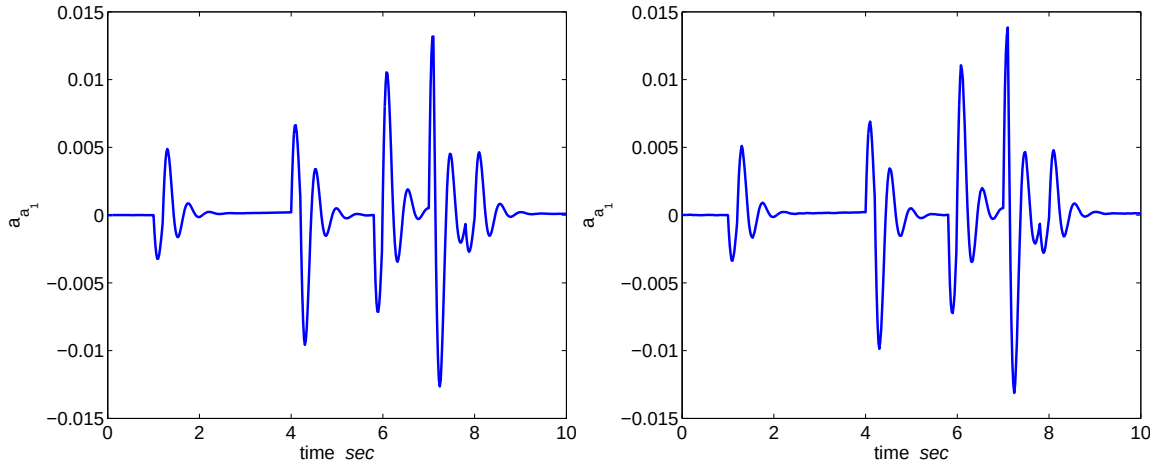

Figure 4.56: $\dot{a}_1$ forward 10m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

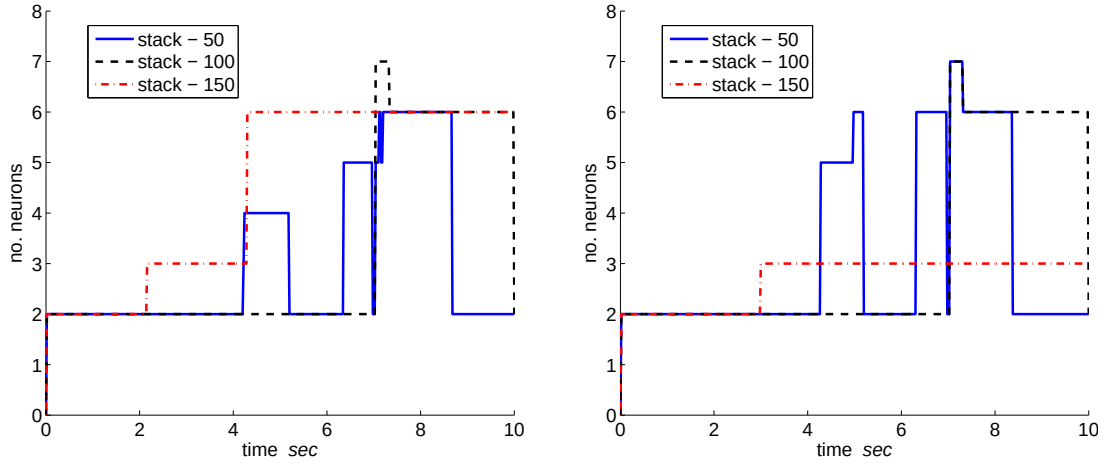

Figure 4.57: $\dot{a}_1$ forward 10m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

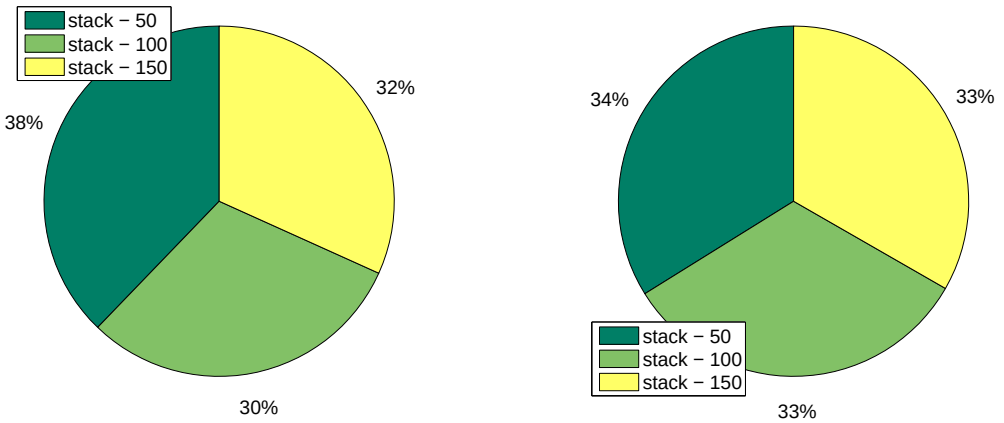

Figure 4.58: $\dot{a}_1$ forward 10m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

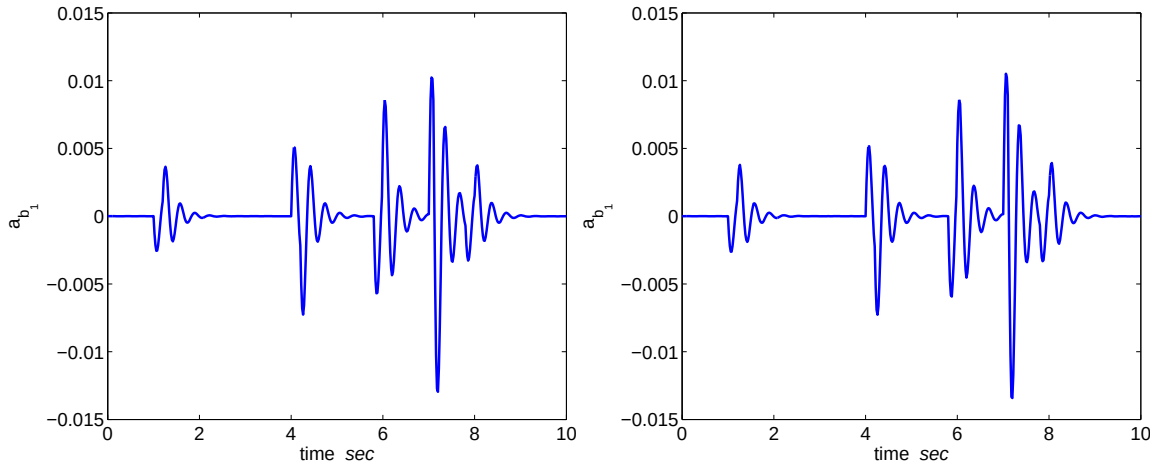

Figure 4.59: $\dot{b}_1$ forward 10m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

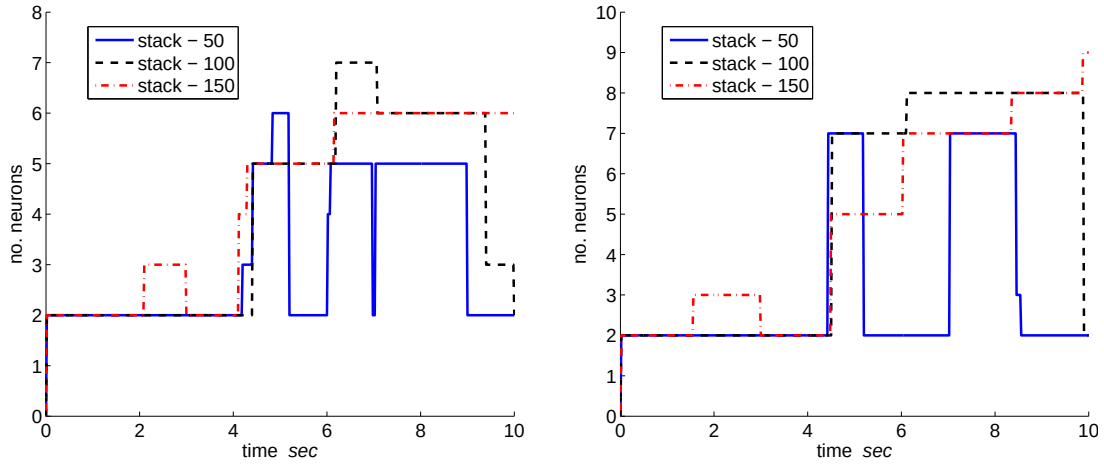

Figure 4.60: $\dot{b}_1$ forward 10m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

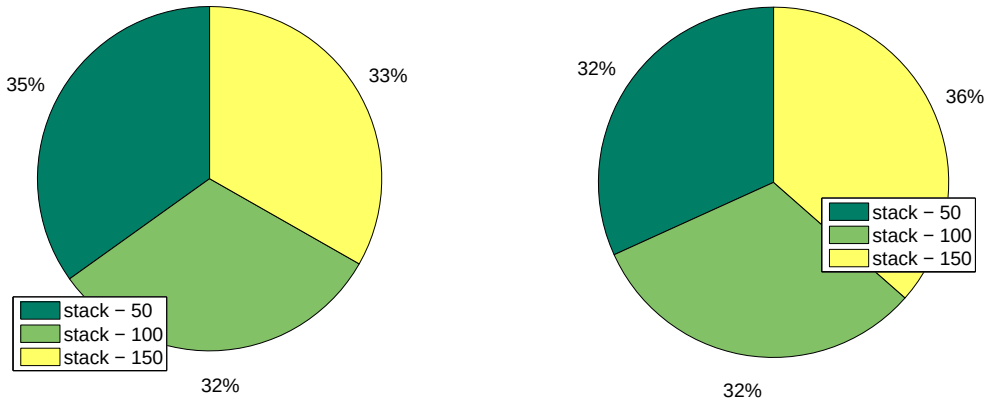

Figure 4.61: $\dot{b}_1$ forward 10m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

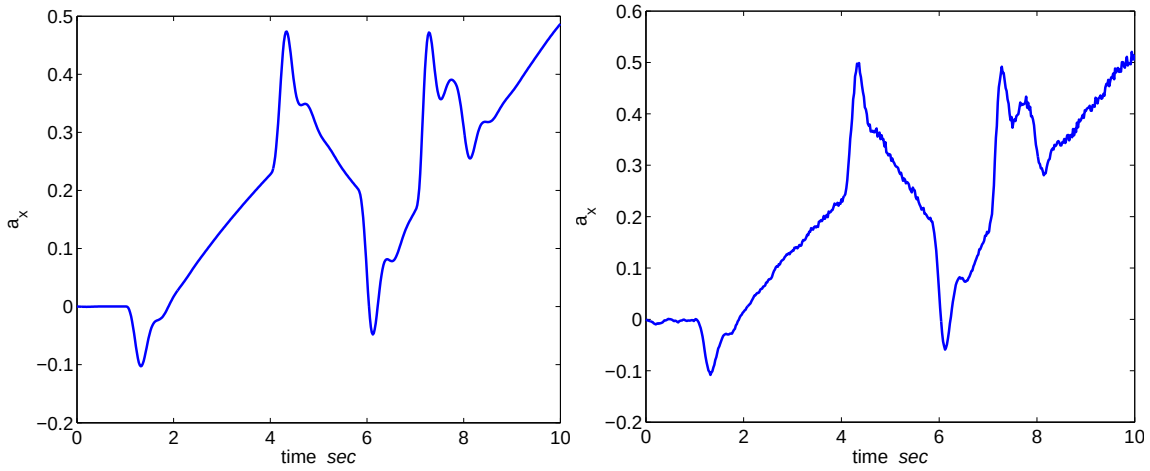

Figure 4.62: a_x forward 20m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

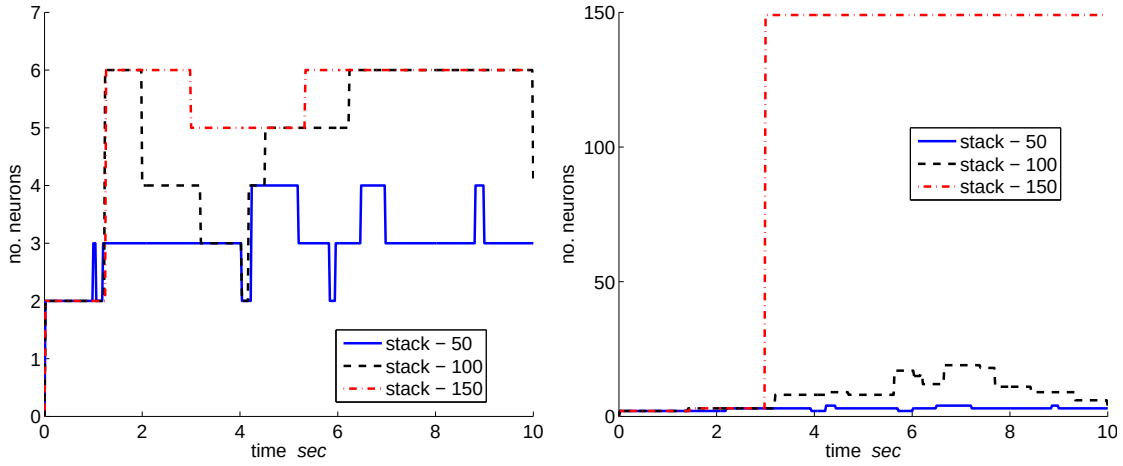

Figure 4.63: a_x forward 20m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

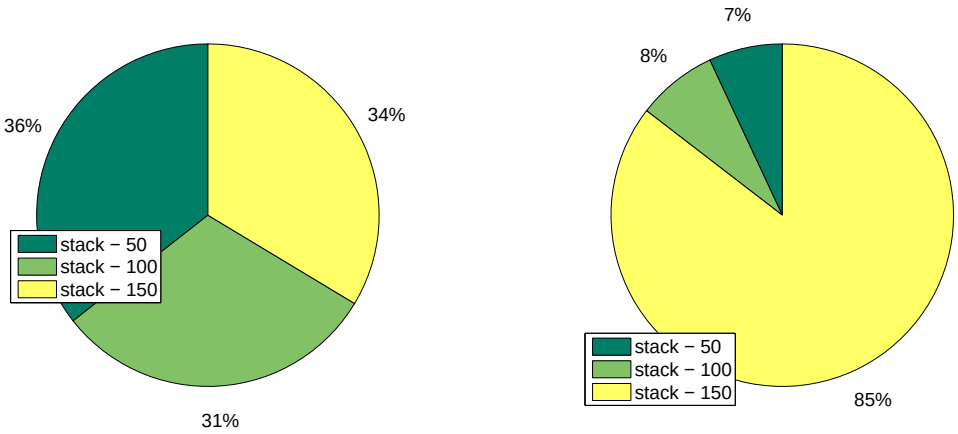

Figure 4.64: a_x forward 20m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

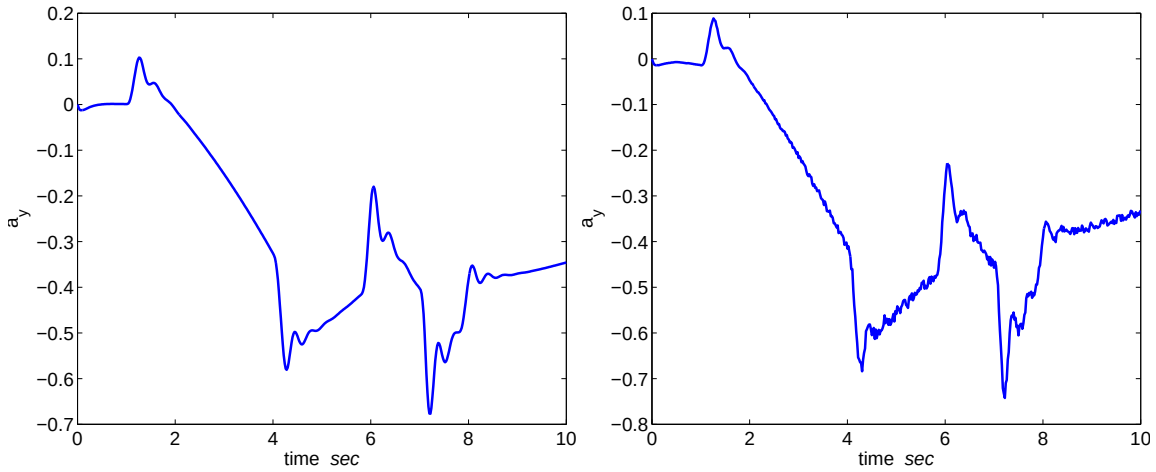

Figure 4.65: a_y forward 20m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

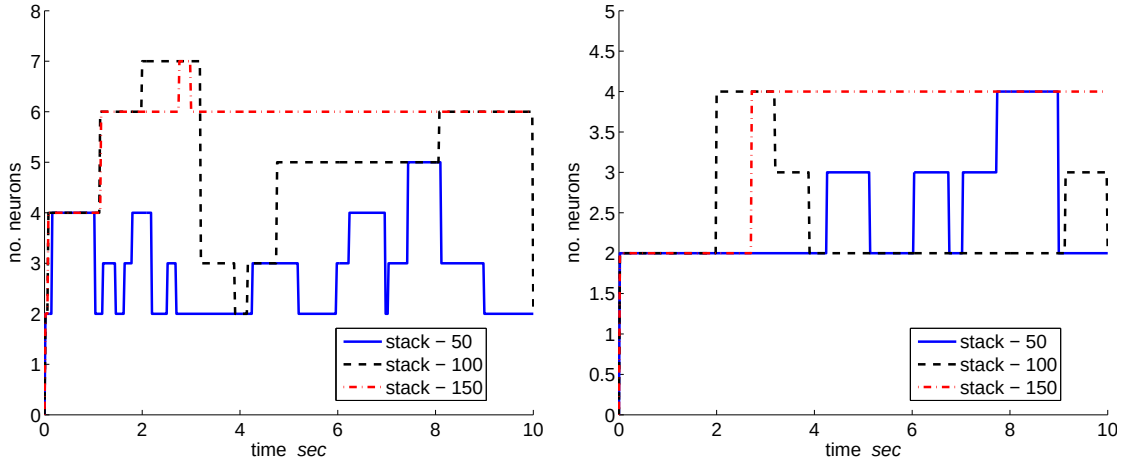

Figure 4.66: a_y forward 20m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

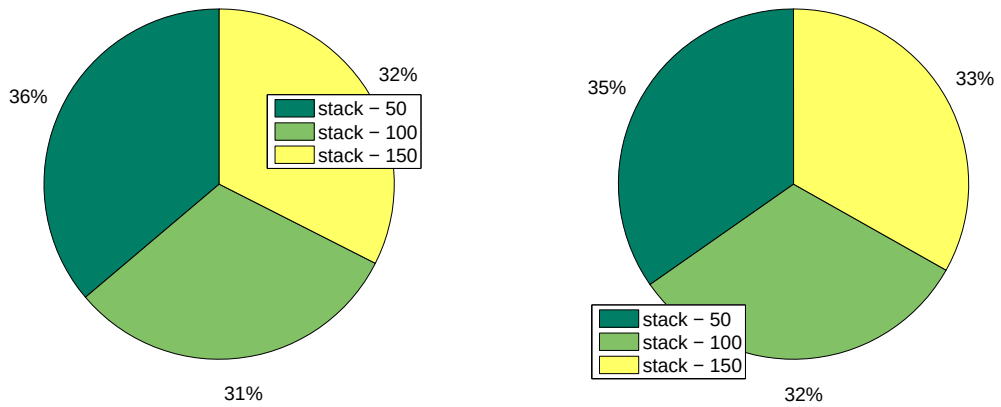

Figure 4.67: a_y forward 20m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

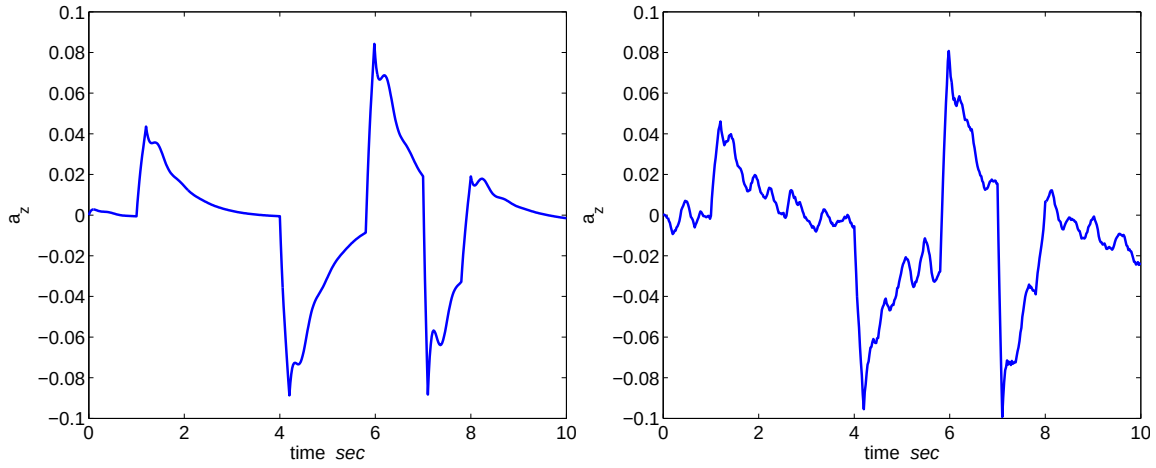

Figure 4.68: a_z forward 20m/s flight online identification using δ_{col}: Time history (1) ideal, (2) turbulence.

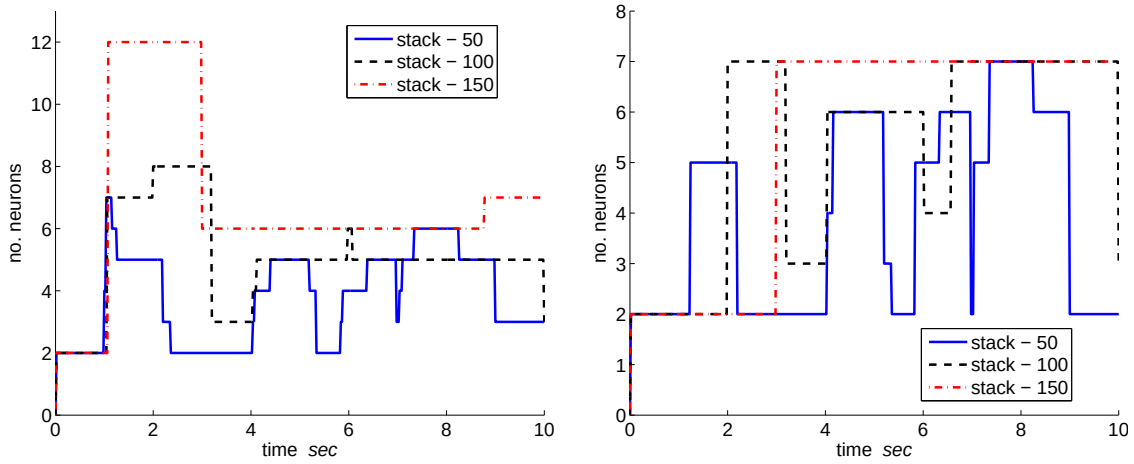

Figure 4.69: a_z forward 20m/s flight online identification using δ_{col}: Network neurons (1) ideal, (2) turbulence.

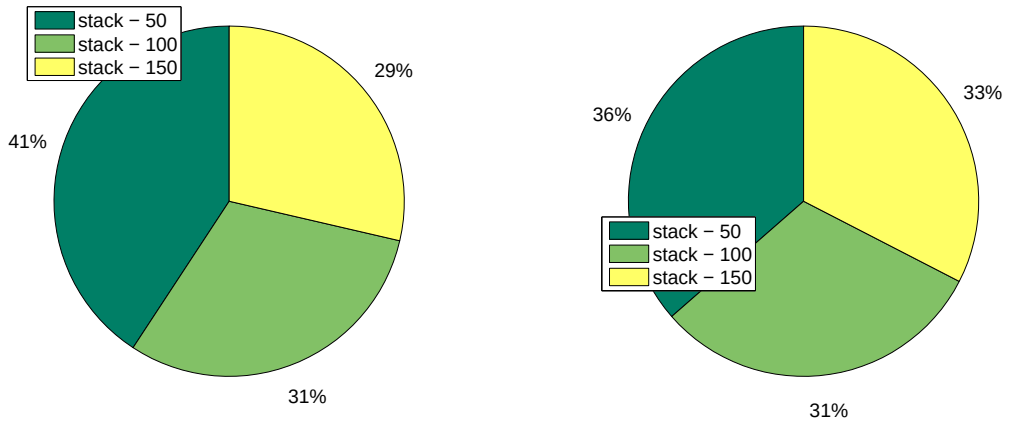

Figure 4.70: a_z forward 20m/s flight online identification using δ_{col}: Memory usage (1) ideal, (2) turbulence.

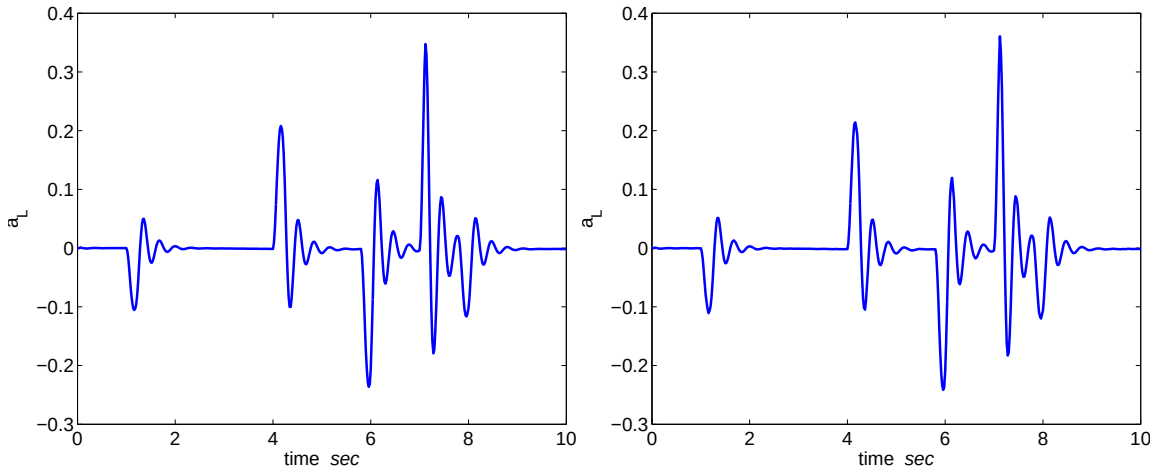

Figure 4.71: $\dot{p}$ forward 20m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

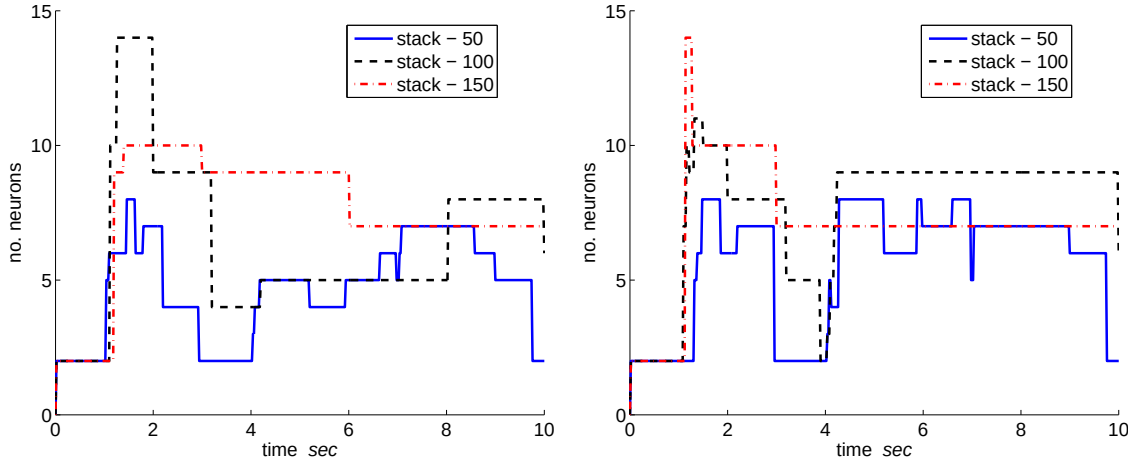

Figure 4.72: $\dot{p}$ forward 20m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

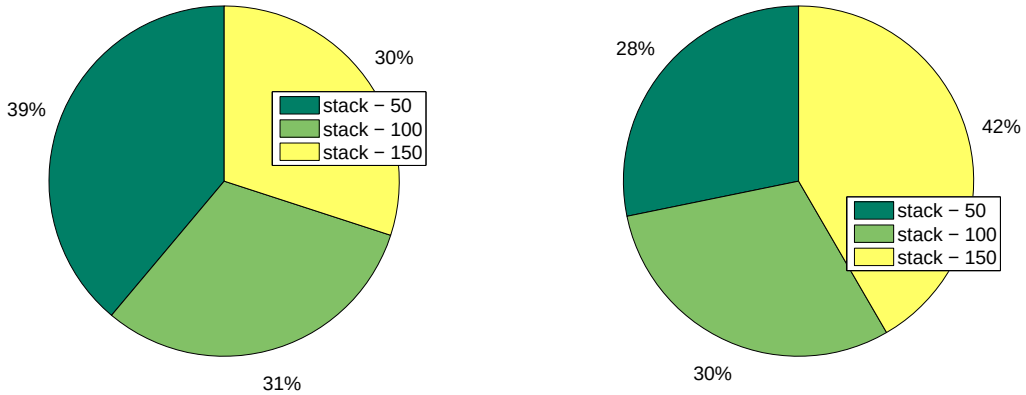

Figure 4.73: $\dot{p}$ forward 20m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

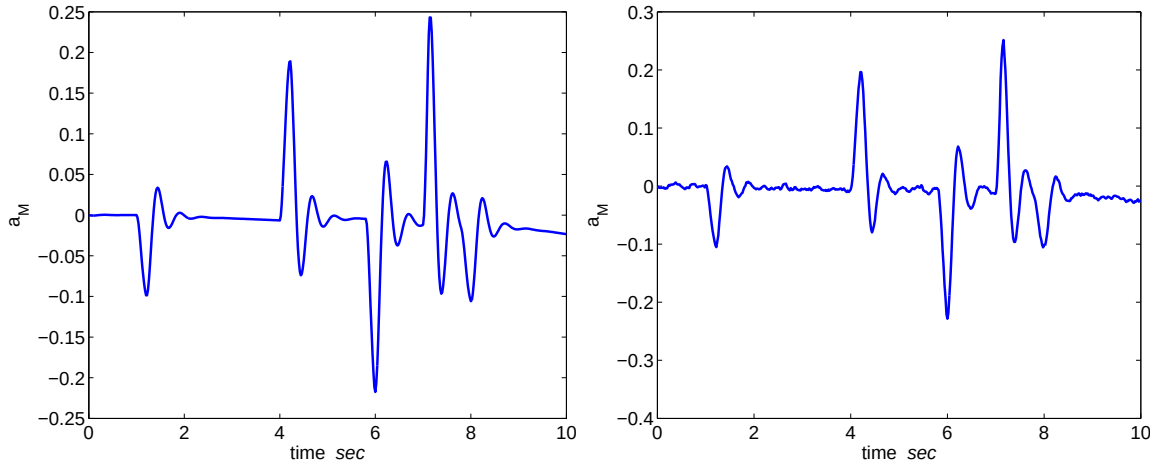

Figure 4.74: $\dot{q}$ forward 20m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

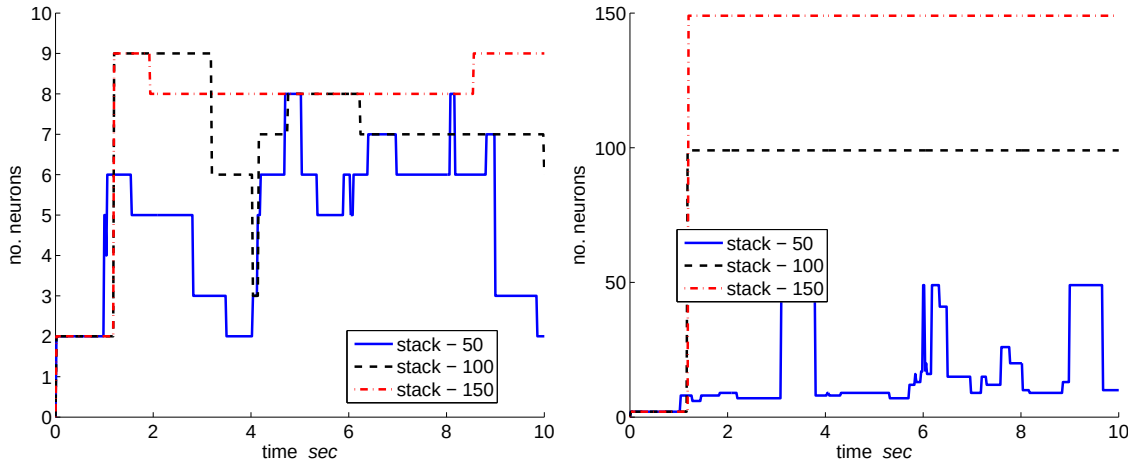

Figure 4.75: $\dot{q}$ forward 20m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

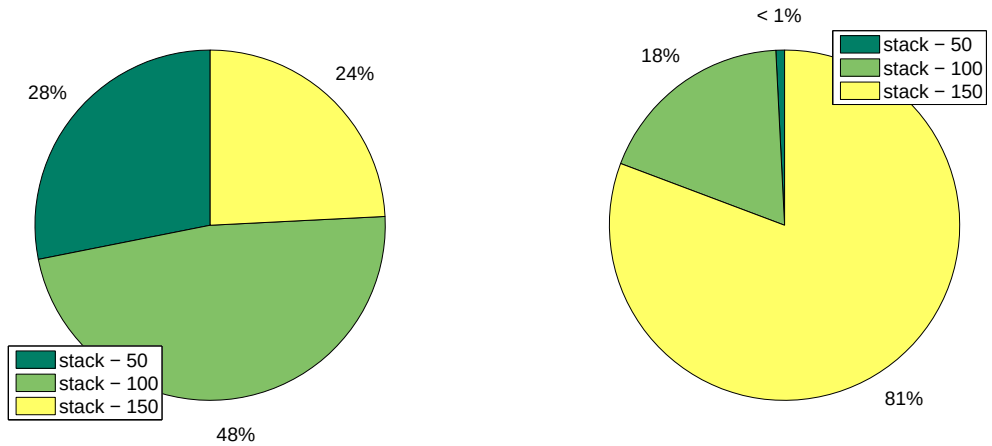

Figure 4.76: $\dot{q}$ forward 20m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

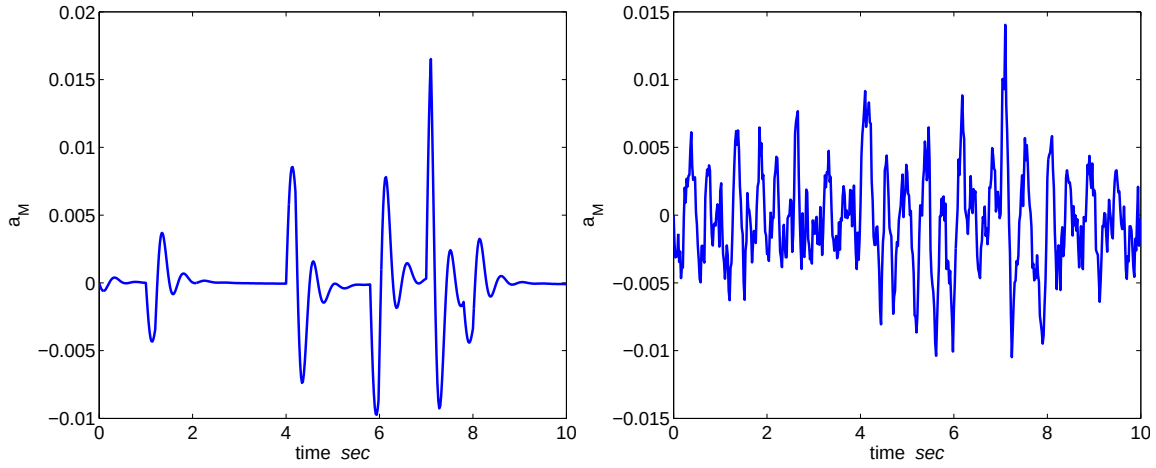

Figure 4.77: $\dot{q}$ forward 20m/s flight online identification using δ_{col}: Time history (1) ideal, (2) turbulence.

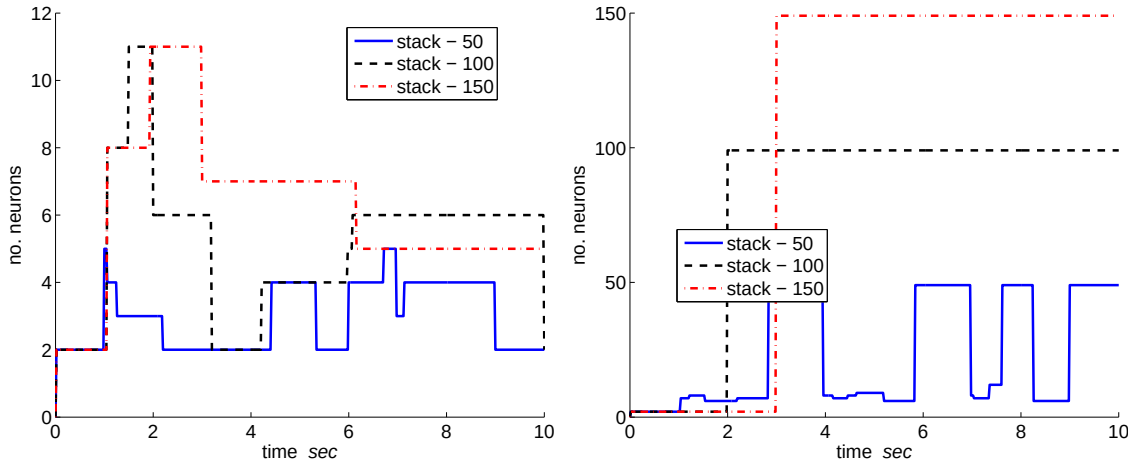

Figure 4.78: $\dot{q}$ forward 20m/s flight online identification using δ_{col}: Network neurons (1) ideal, (2) turbulence.

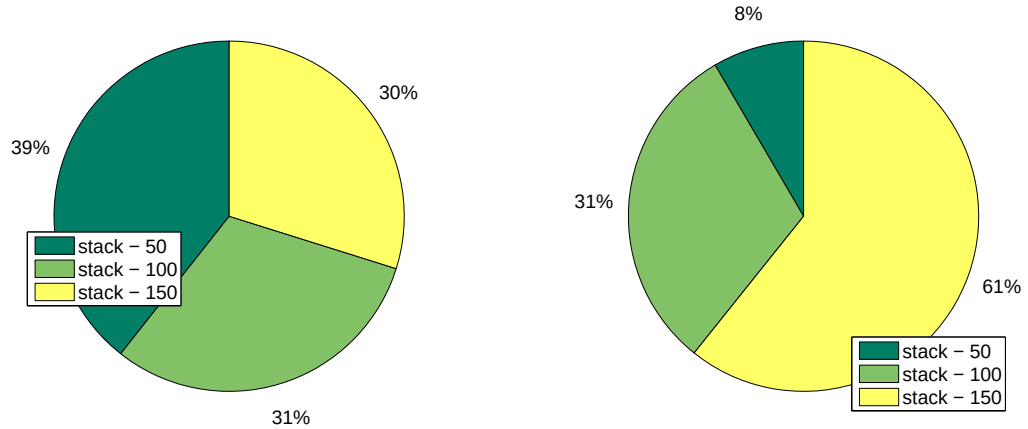

Figure 4.79: $\dot{q}$ forward 20m/s flight online identification using δ_{col}: Memory usage (1) ideal, (2) turbulence.

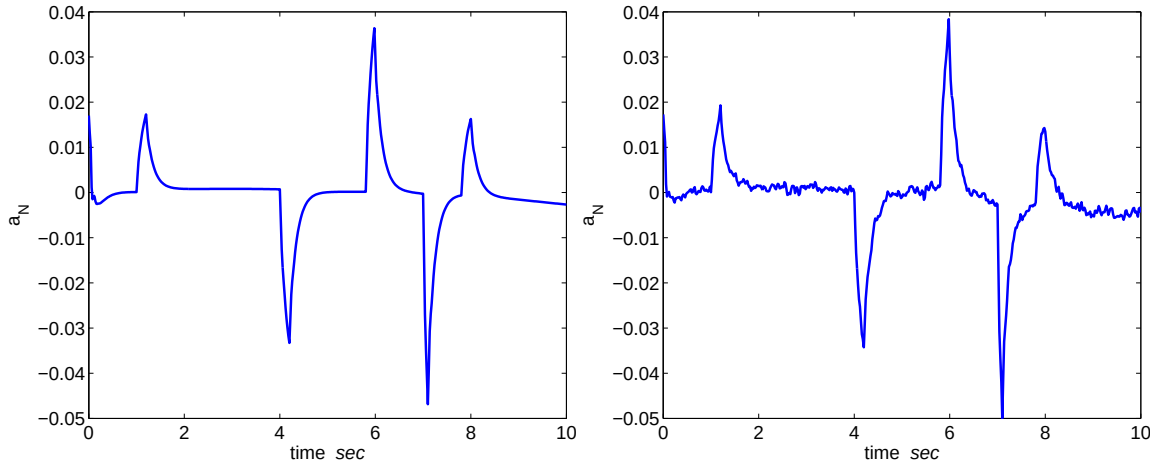

Figure 4.80: $\dot{r}$ forward 20m/s flight online identification using δ_r: Time history (1) ideal, (2) turbulence.

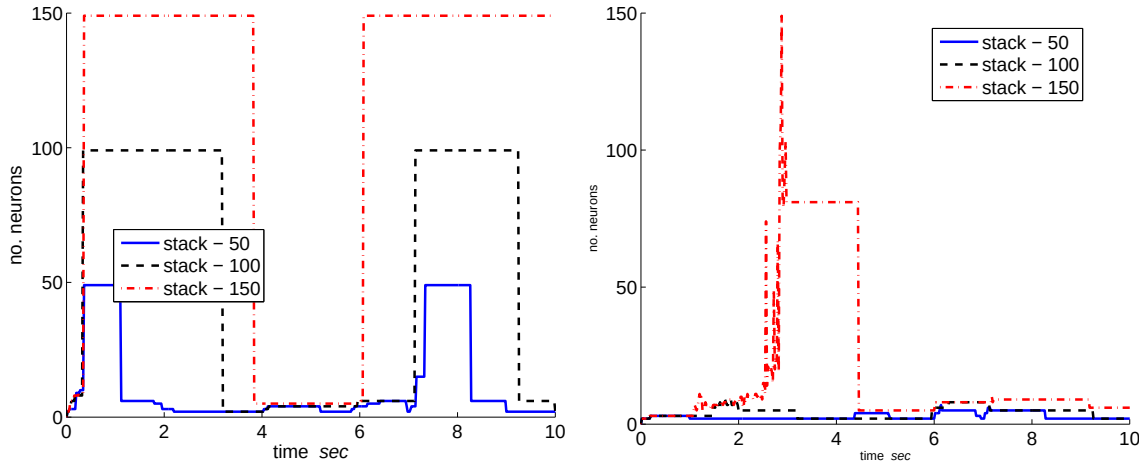

Figure 4.81: $\dot{r}$ forward 20m/s flight online identification using δ_r: Network neurons (1) ideal, (2) turbulence.

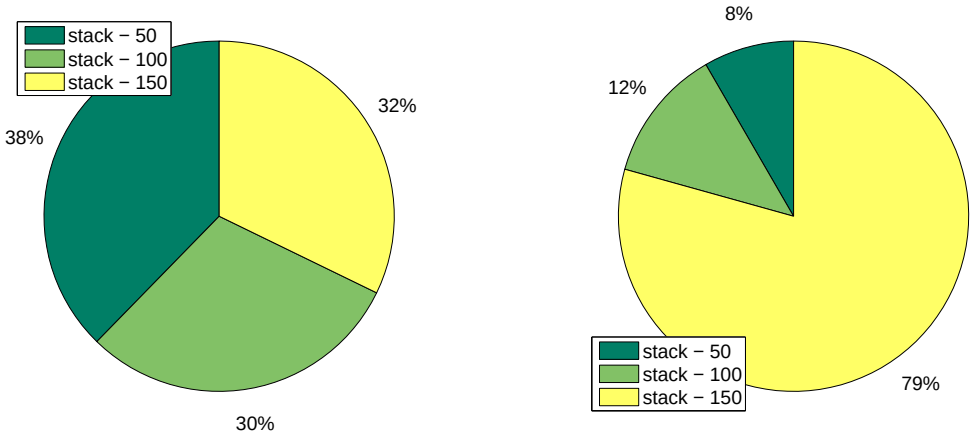

Figure 4.82: $\dot{r}$ forward 20m/s flight online identification using δ_r: Memory usage (1) ideal, (2) turbulence.

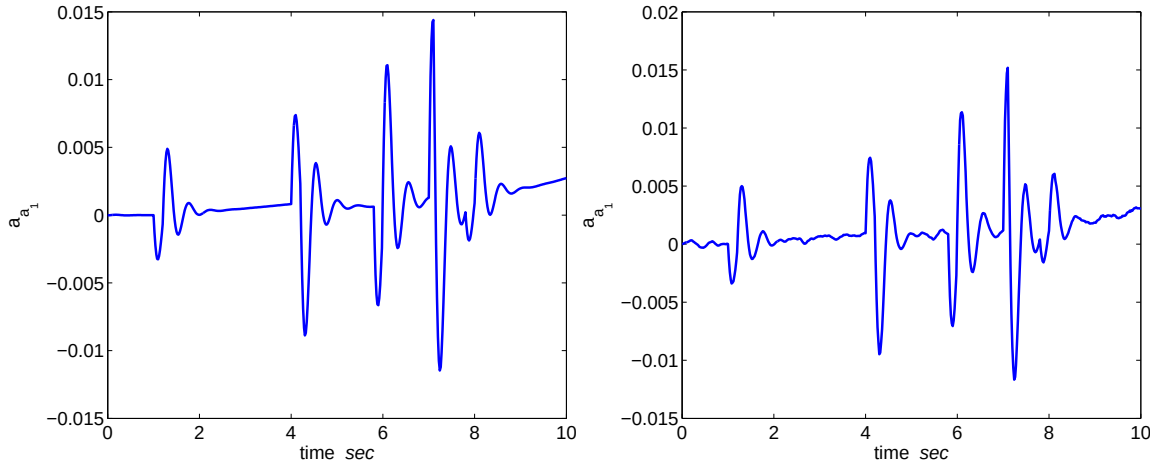

Figure 4.83: $\dot{a}_1$ forward 20m/s flight online identification using δ_{lon}: Time history (1) ideal, (2) turbulence.

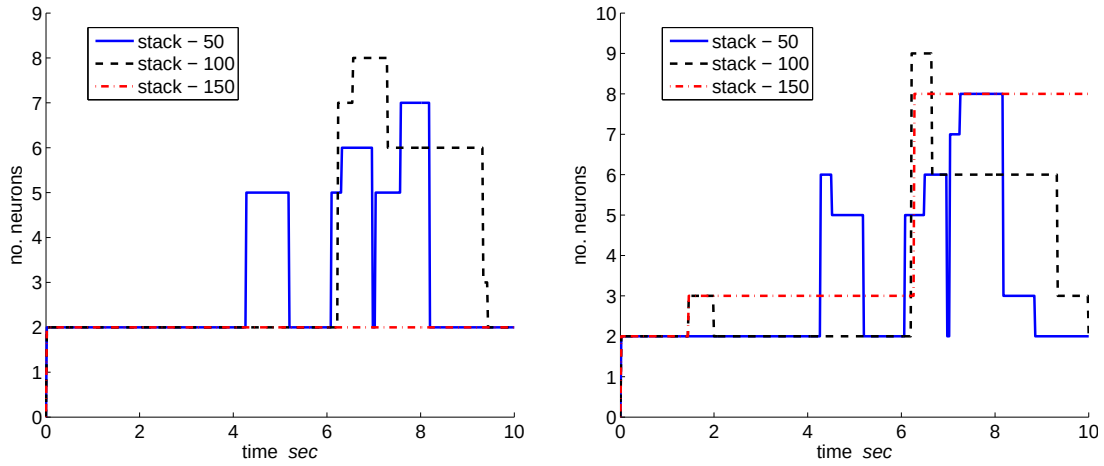

Figure 4.84: $\dot{a}_1$ forward 20m/s flight online identification using δ_{lon}: Network neurons (1) ideal, (2) turbulence.

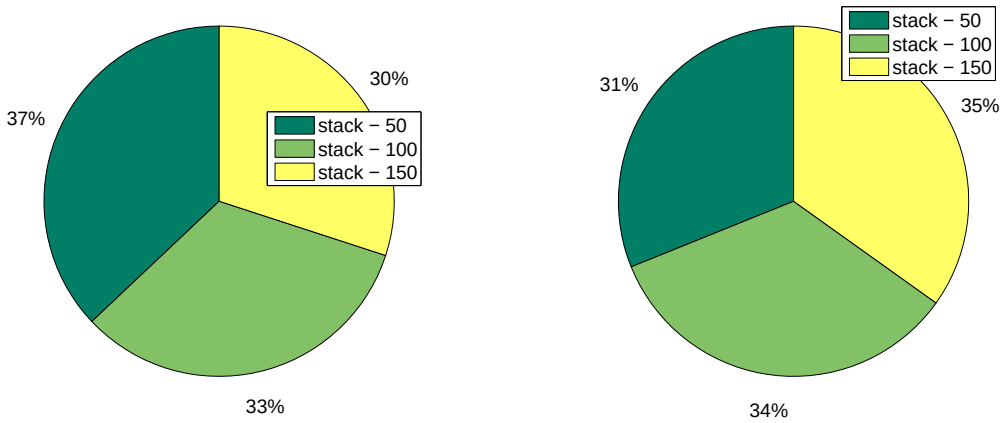

Figure 4.85: $\dot{a}_1$ forward 20m/s flight online identification using δ_{lon}: Memory usage (1) ideal, (2) turbulence.

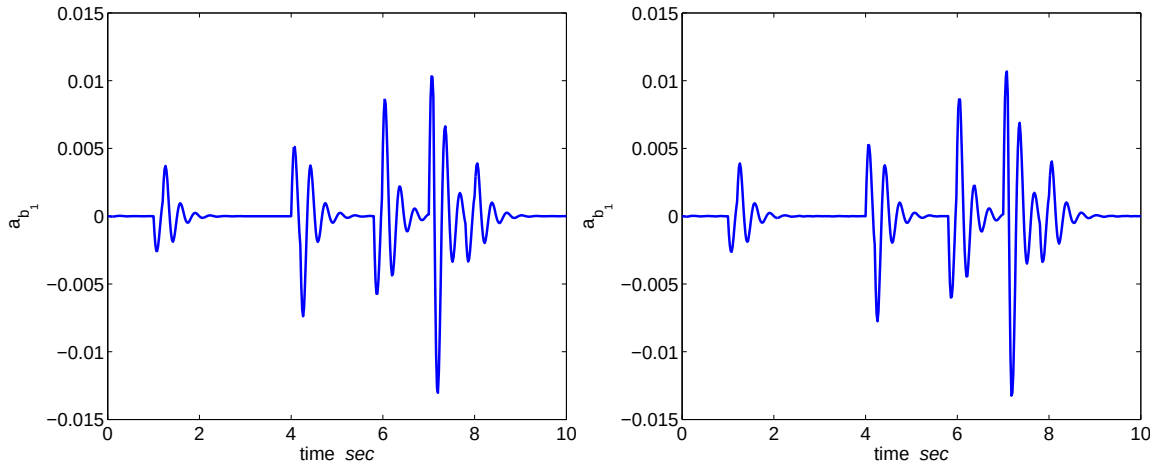

Figure 4.86: $\dot{b}_1$ forward 20m/s flight online identification using δ_{lat}: Time history (1) ideal, (2) turbulence.

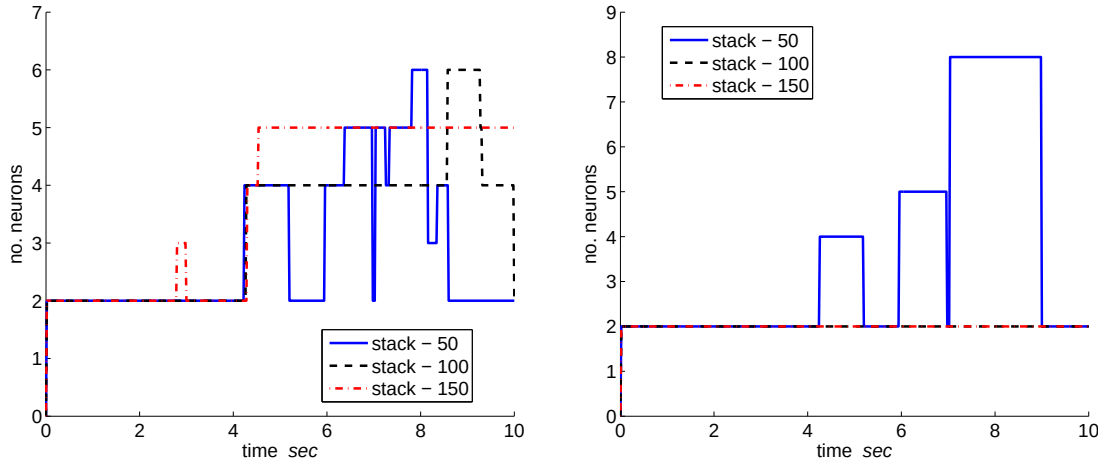

Figure 4.87: $\dot{b}_1$ forward 20m/s flight online identification using δ_{lat}: Network neurons (1) ideal, (2) turbulence.

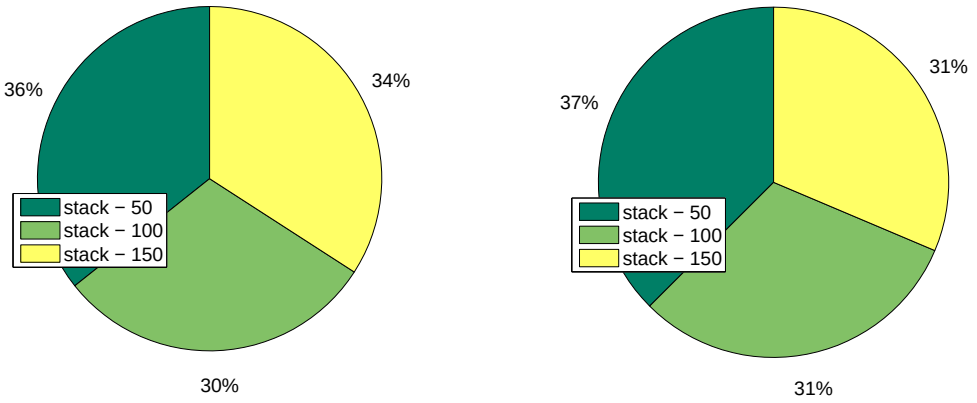

Figure 4.88: $\dot{b}_1$ forward 20m/s flight online identification using δ_{lat}: Memory usage (1) ideal, (2) turbulence.

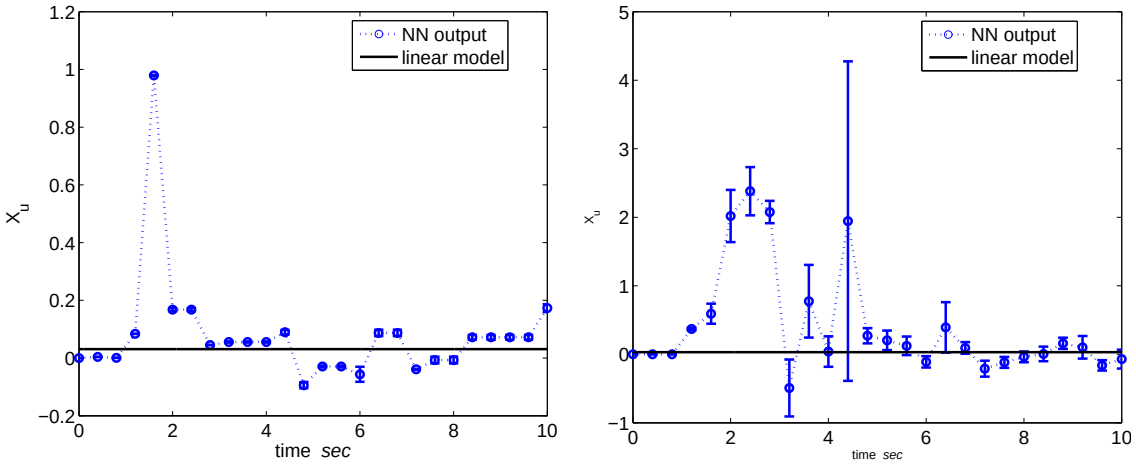

Figure 4.89: X_u hover flight DM online estimation: (1) ideal, (2) turbulence.

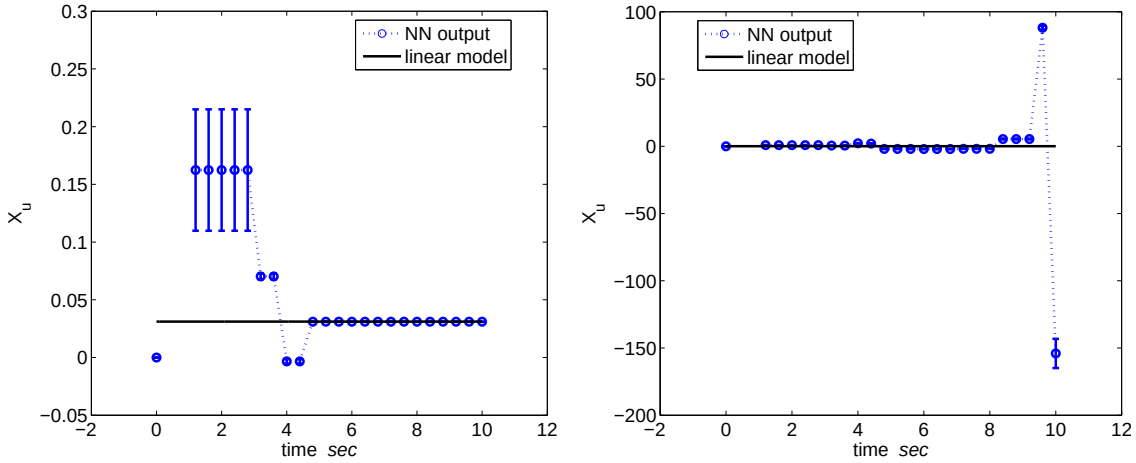

Figure 4.90: X_u hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.2: X_u hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(34)	2(21)	0(1)
stack - 100	0(6)	0(11)	3(34)	0(2)
stack - 150	0(8)	265(265)	4(22)	0(2)

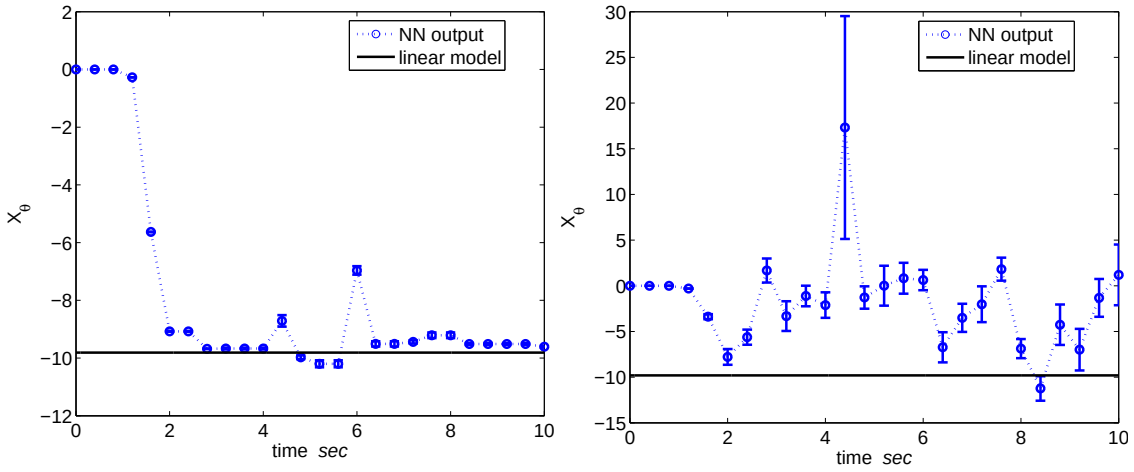

Figure 4.91: X_θ hover flight DM online estimation: (1) ideal, (2) turbulence.

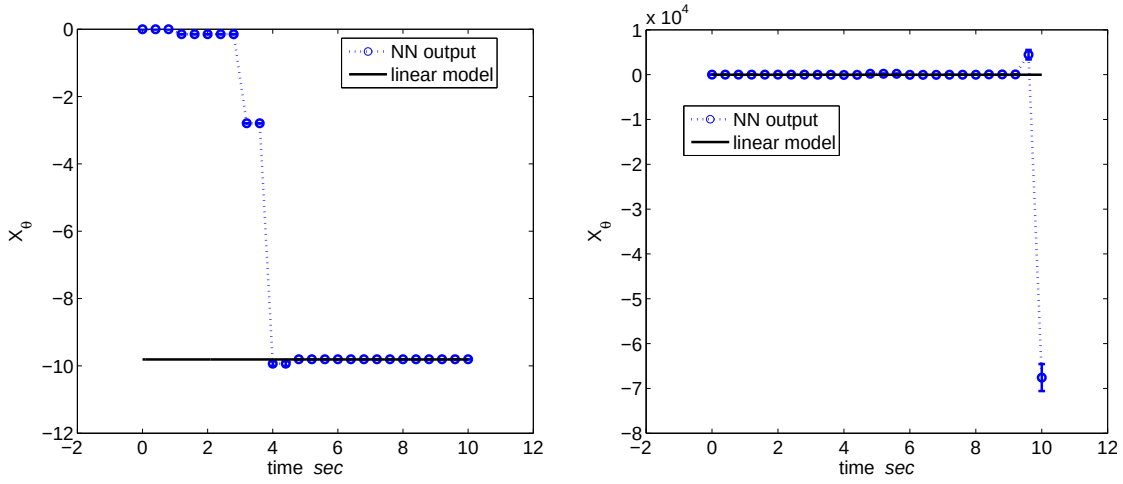

Figure 4.92: X_θ hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.3: X_θ hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	73(208)	42(122)	5(61)	0(0)
stack - 100	199(349)	0(284)	5(70)	0(45)
stack - 150	279(404)	320(320)	2(58)	0(0)

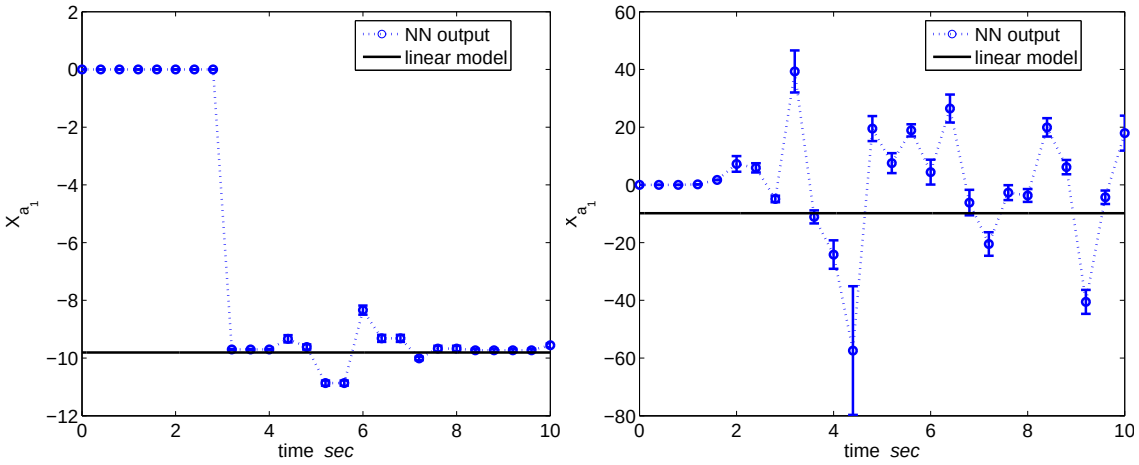

Figure 4.93: X_{a_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

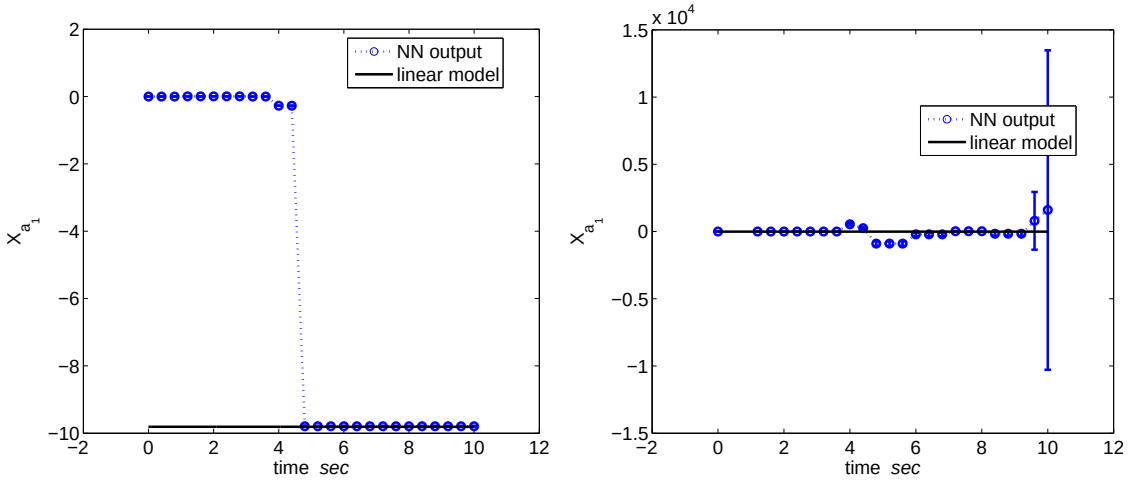

Figure 4.94: X_{a_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.4: X_{a_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	140(243)	39(90)	8(55)	0(0)
stack - 100	238(356)	0(0)	9(59)	0(45)
stack - 150	255(359)	265(265)	4(67)	0(2)

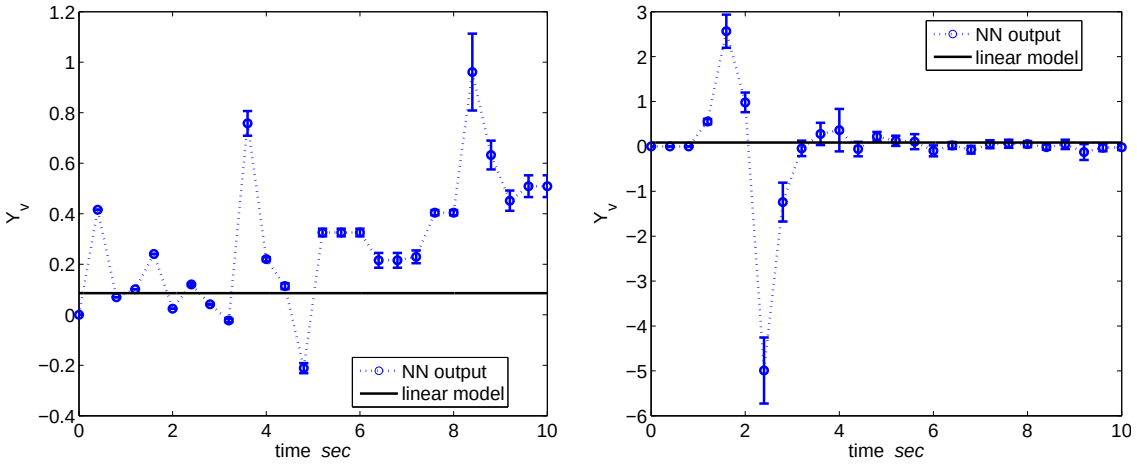

Figure 4.95: Y_v hover flight DM online estimation: (1) ideal, (2) turbulence.

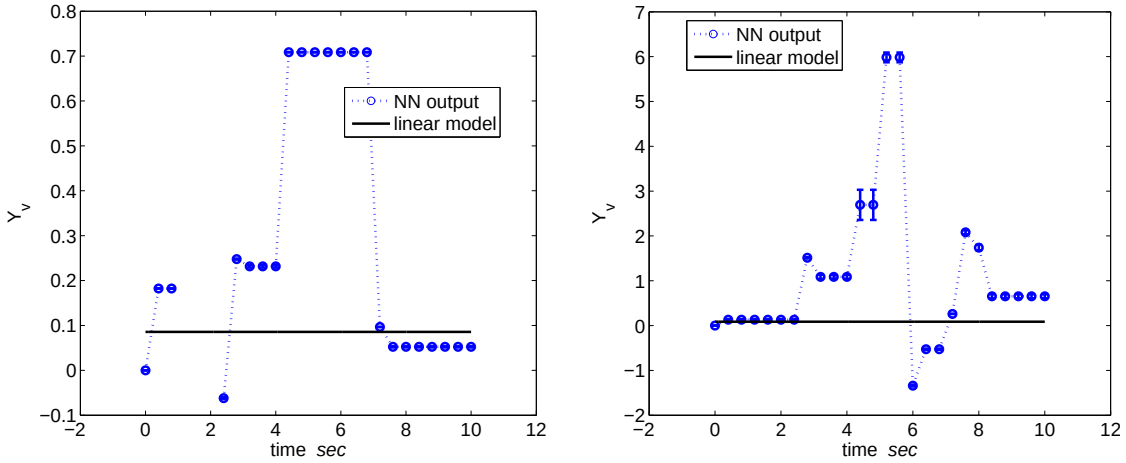

Figure 4.96: Y_v hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.5: Y_v hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(23)	0(8)	7(38)	0(0)
stack - 100	0(10)	0(171)	6(56)	0(0)
stack - 150	0(44)	0(140)	8(67)	0(0)

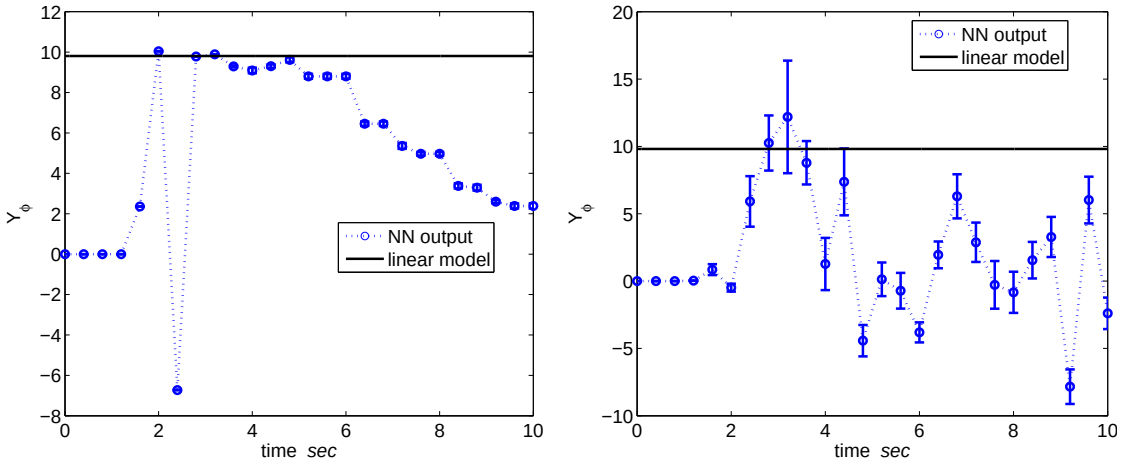

Figure 4.97: Y_ϕ hover flight DM online estimation: (1) ideal, (2) turbulence.

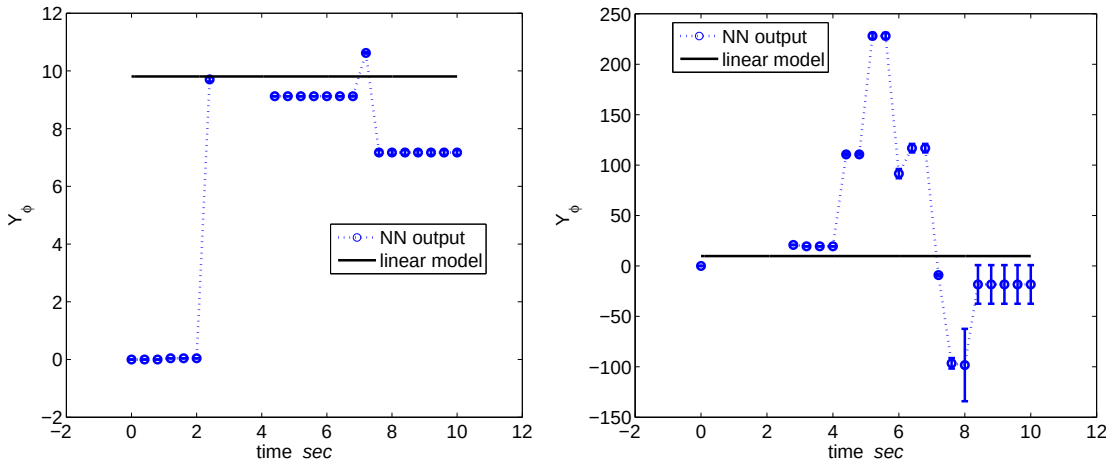

Figure 4.98: Y_ϕ hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.6: Y_ϕ hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(67)	0(88)	14(86)	0(0)
stack - 100	84(143)	24(305)	9(82)	0(78)
stack - 150	56(218)	25(306)	11(85)	0(0)

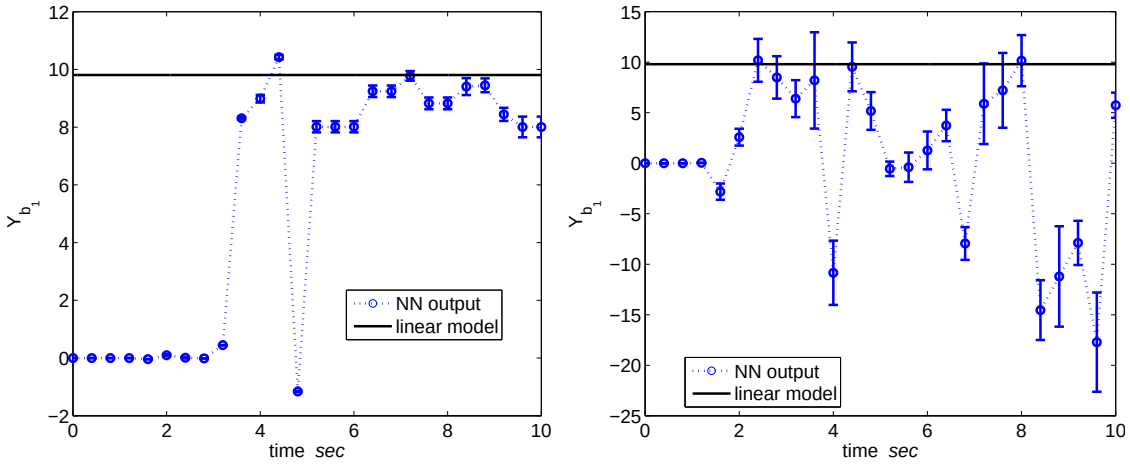

Figure 4.99: Y_{b_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

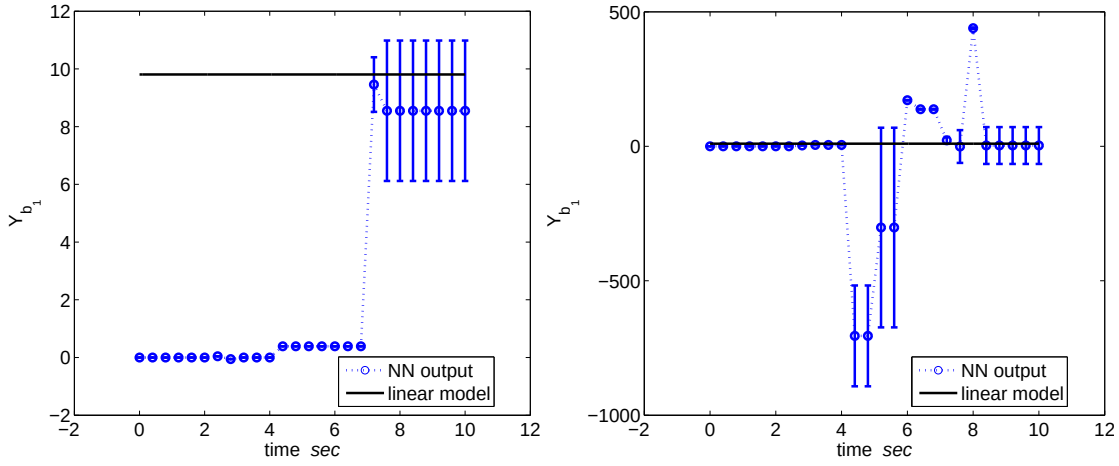

Figure 4.100: Y_{b_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.7: Y_{b_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	39(167)	0(79)	8(49)	0(12)
stack - 100	129(307)	0(171)	7(61)	0(0)
stack - 150	48(298)	7(140)	10(63)	0(0)

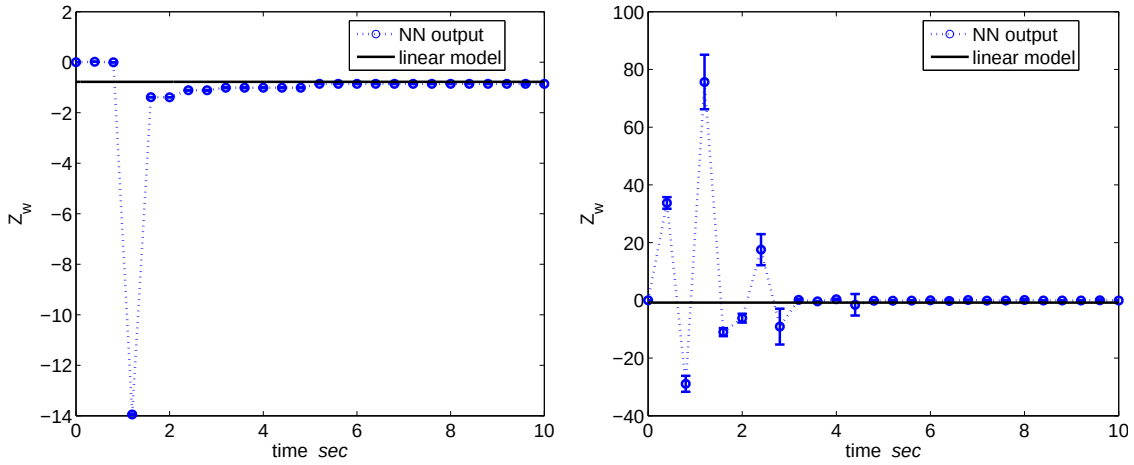

Figure 4.101: Z_w hover flight DM online estimation: (1) ideal, (2) turbulence.

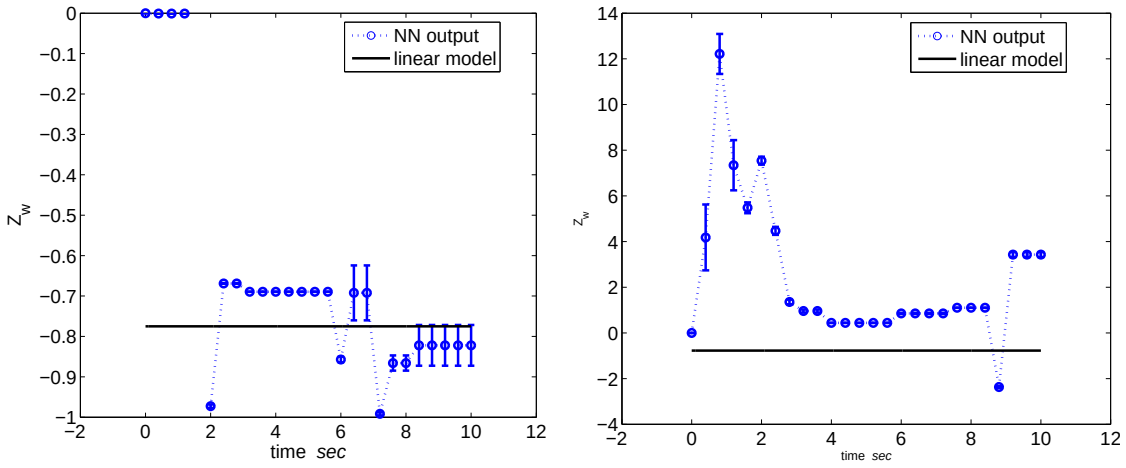

Figure 4.102: Z_w hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.8: Z_w hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	92(376)	0(61)	7(21)	0(0)
stack - 100	25(280)	98(301)	3(20)	1(2)
stack - 150	0(351)	6(414)	2(12)	0(0)

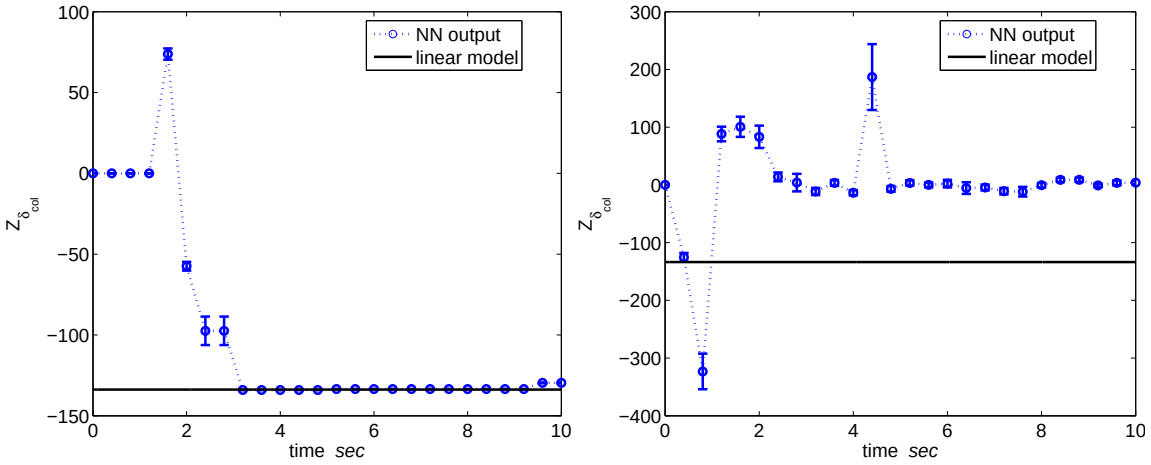

Figure 4.103: $Z_{\delta_{col}}$ hover flight DM online estimation: (1) ideal, (2) turbulence.

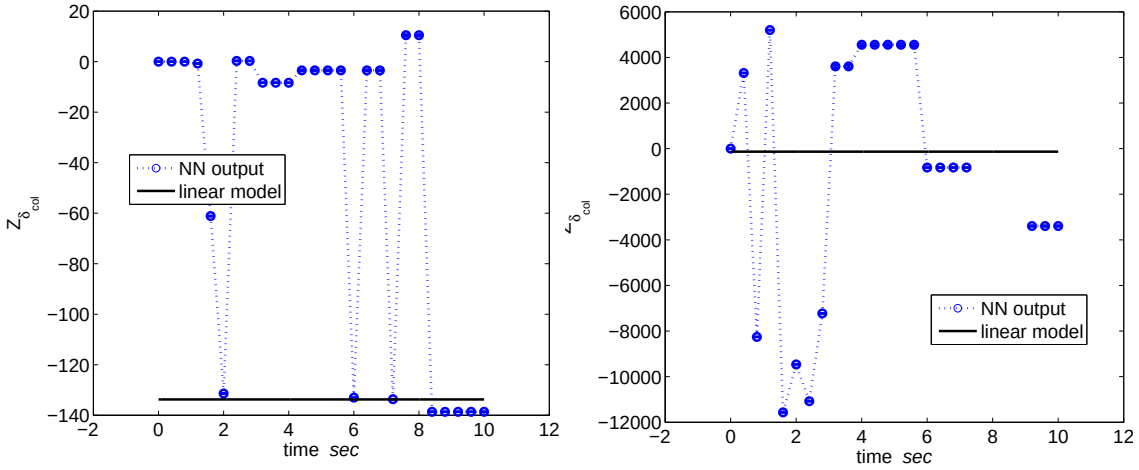

Figure 4.104: $Z_{\delta_{col}}$ hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.9: $Z_{\delta_{col}}$ hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	292(292)	165(176)	1(3)	0(0)
stack - 100	404(404)	0(0)	1(18)	0(0)
stack - 150	351(397)	140(140)	1(16)	0(0)

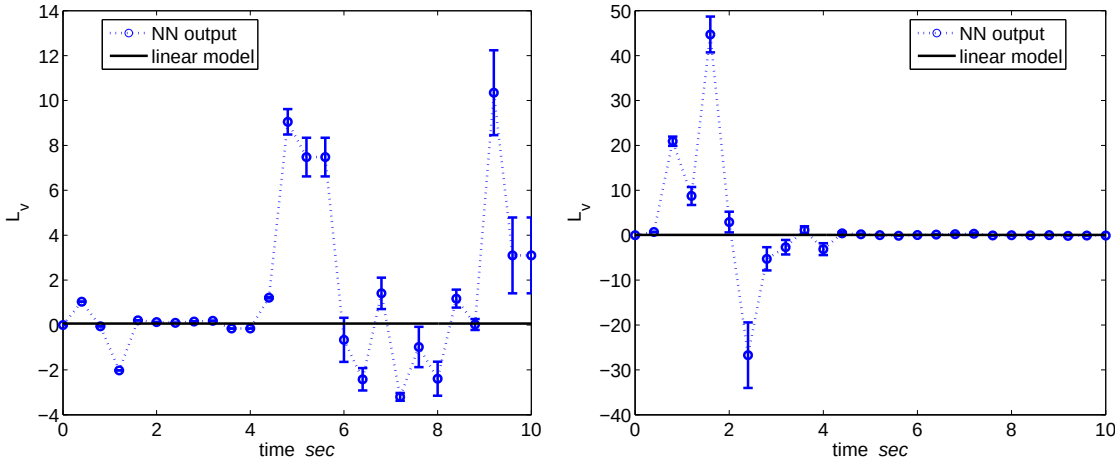

Figure 4.105: L_v hover flight DM online estimation: (1) ideal, (2) turbulence.

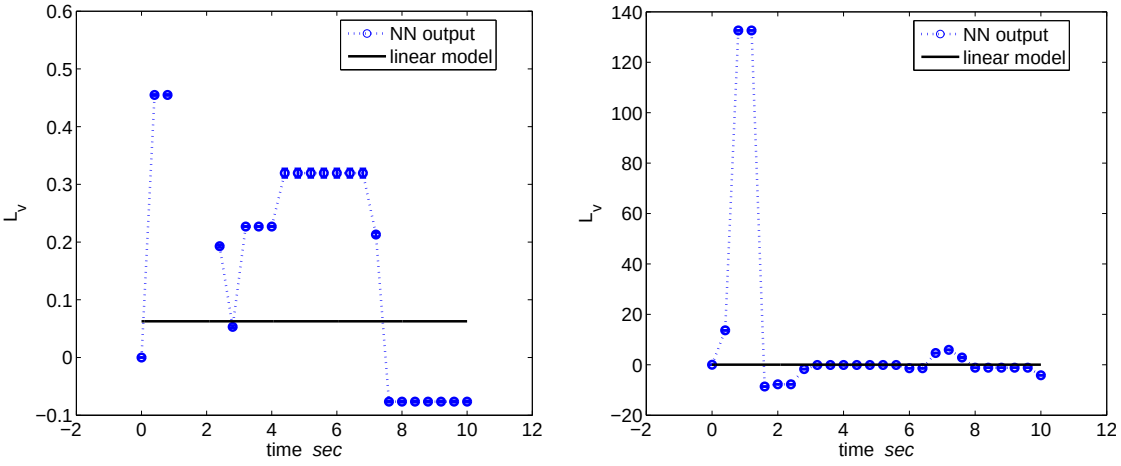

Figure 4.106: L_v hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.10: L_v hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(0)	2(27)	0(0)
stack - 100	0(17)	0(0)	3(32)	0(0)
stack - 150	0(0)	6(23)	5(31)	0(0)

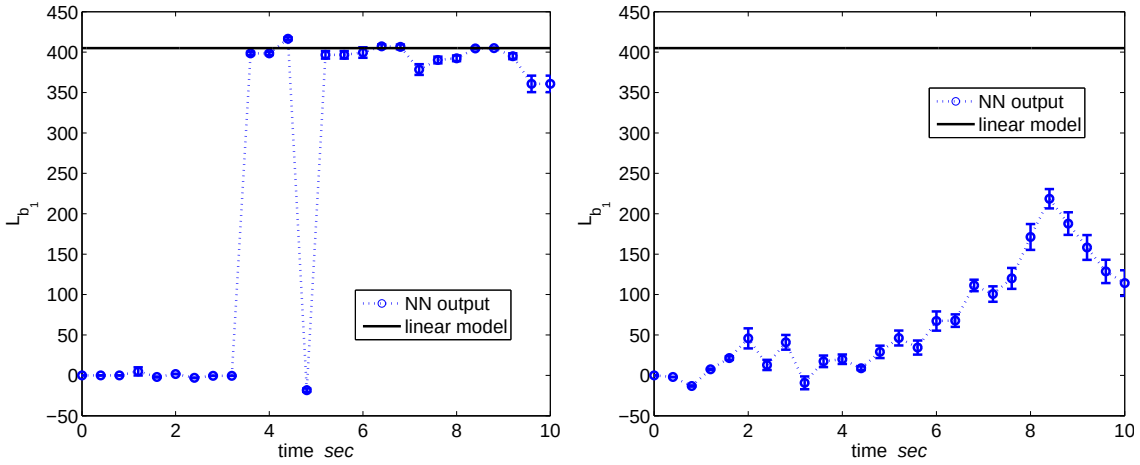

Figure 4.107: L_{b_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

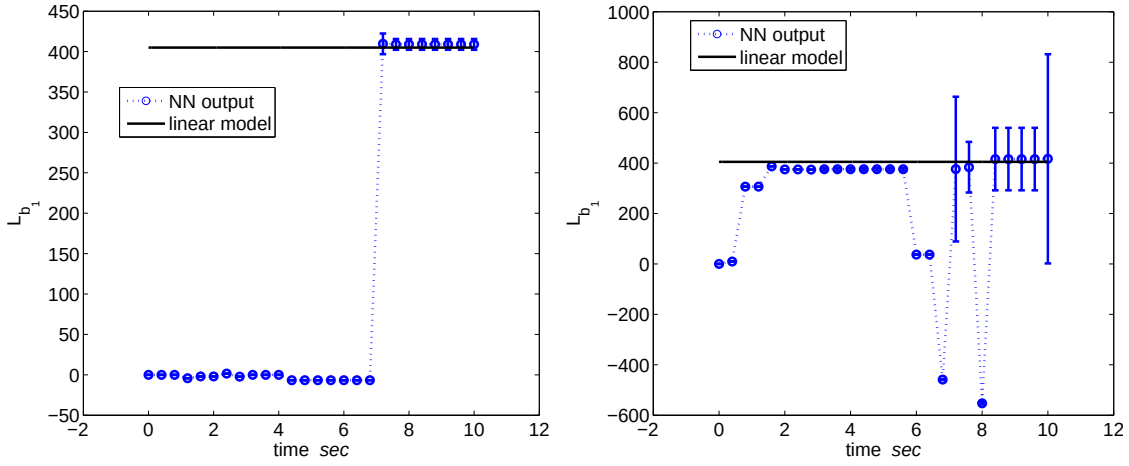

Figure 4.108: L_{b_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.11: L_{b_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	204(231)	6(6)	0(1)	24(70)
stack - 100	235(313)	171(171)	2(5)	130(244)
stack - 150	239(298)	140(140)	0(3)	116(395)

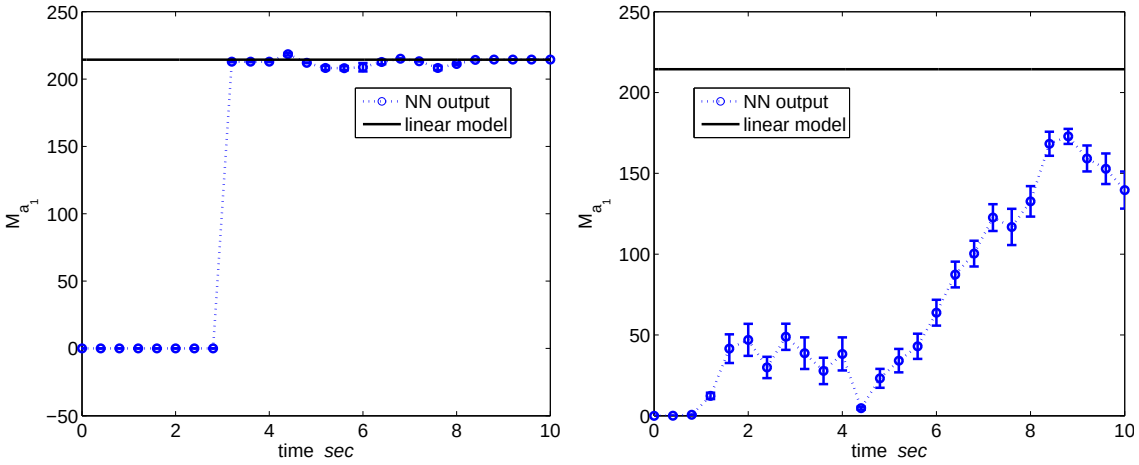

Figure 4.109: M_{a_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

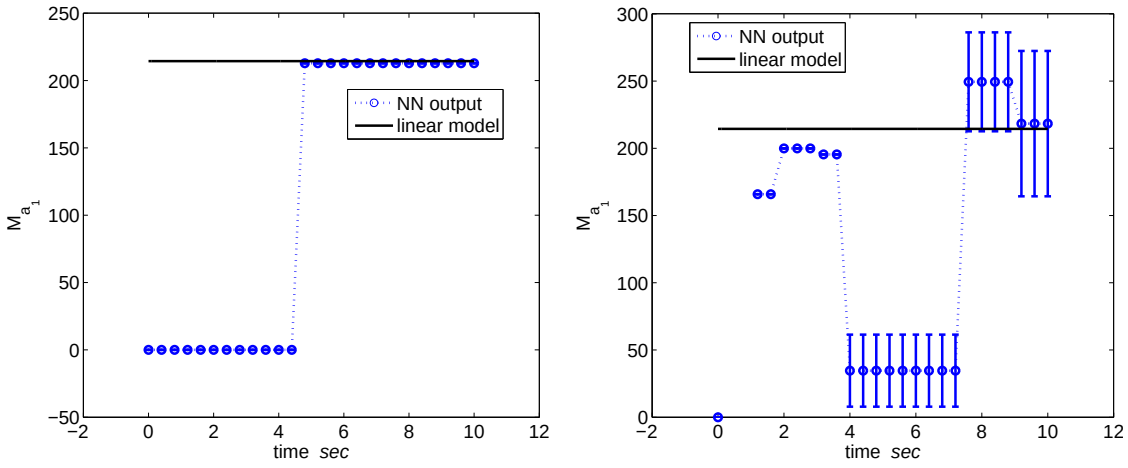

Figure 4.110: M_{a_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.12: M_{a_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	208(261)	51(122)	1(49)	0(79)
stack - 100	345(364)	284(284)	0(119)	2(372)
stack - 150	359(359)	265(265)	0(111)	48(265)

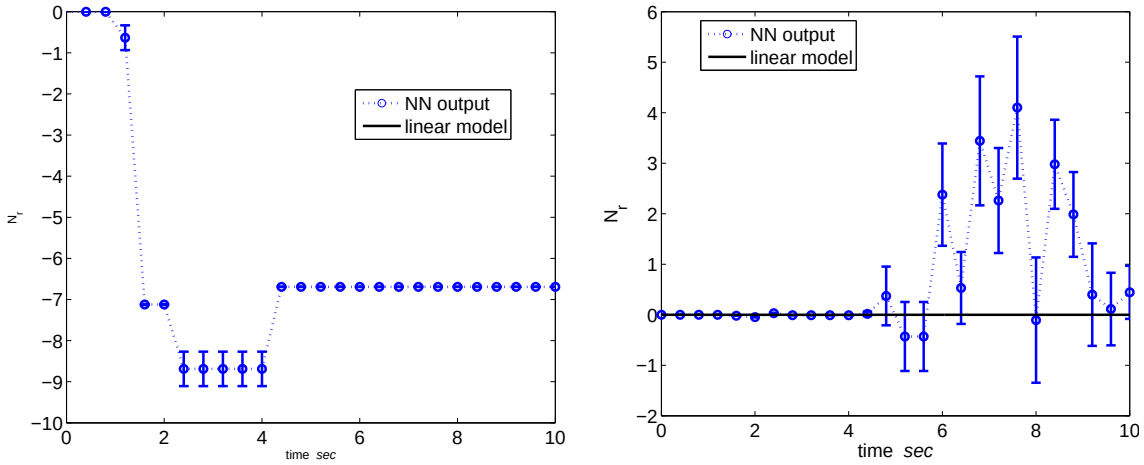

Figure 4.111: N_r hover flight DM online estimation: (1) ideal, (2) turbulence.

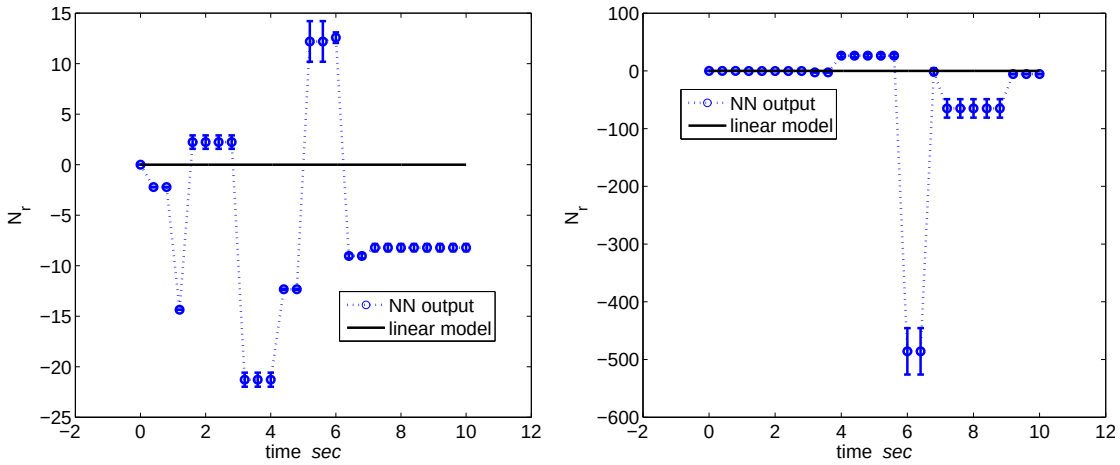

Figure 4.112: N_r hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.13: N_r hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(0)	0(0)	0(0)
stack - 100	0(0)	0(0)	0(0)	0(0)
stack - 150	0(0)	0(0)	0(0)	0(0)

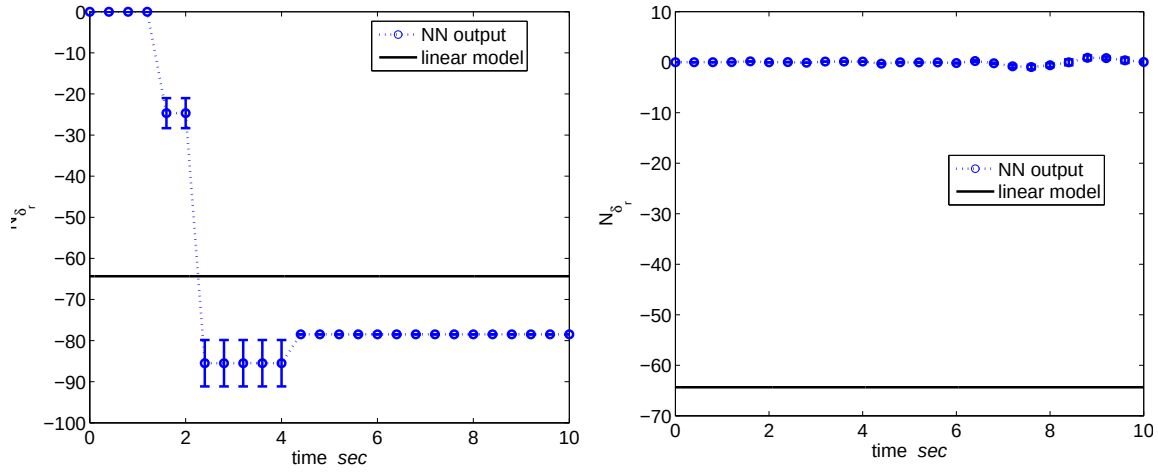

Figure 4.113: N_{δ_r} hover flight DM online estimation: (1) ideal, (2) turbulence.

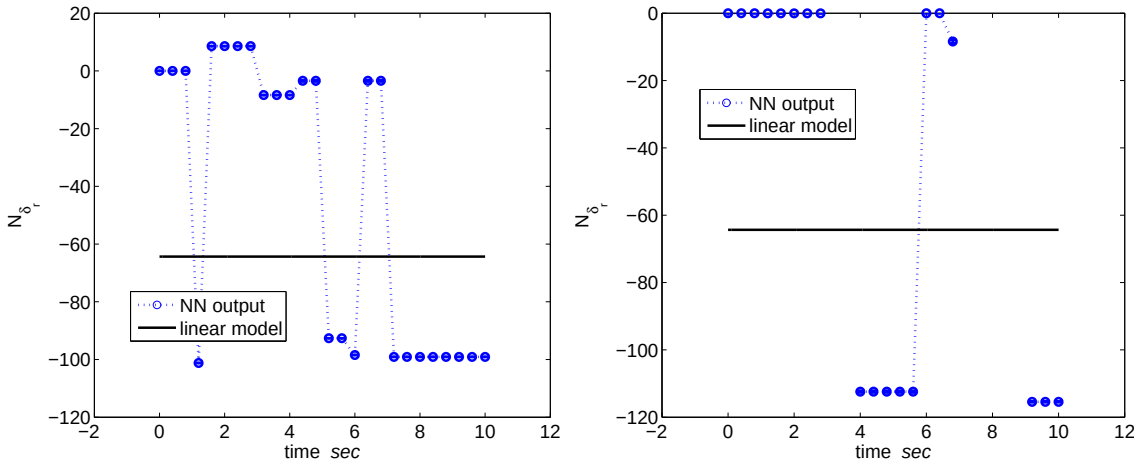

Figure 4.114: N_{δ_r} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.14: N_{δ_r} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(0)	0(0)	0(1)
stack - 100	0(43)	0(0)	0(0)	0(0)
stack - 150	0(384)	0(0)	0(0)	0(0)

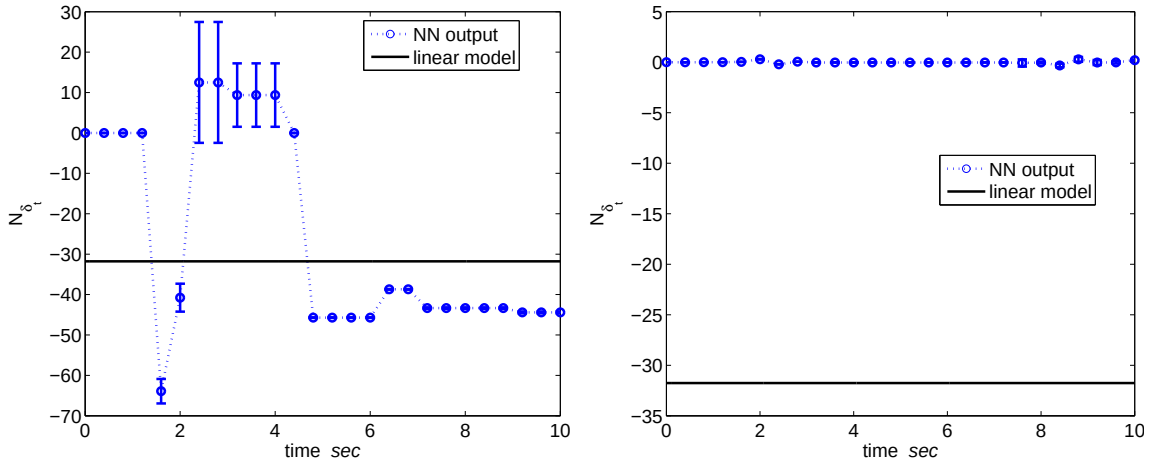

Figure 4.115: N_{δ_t} hover flight DM online estimation: (1) ideal, (2) turbulence.

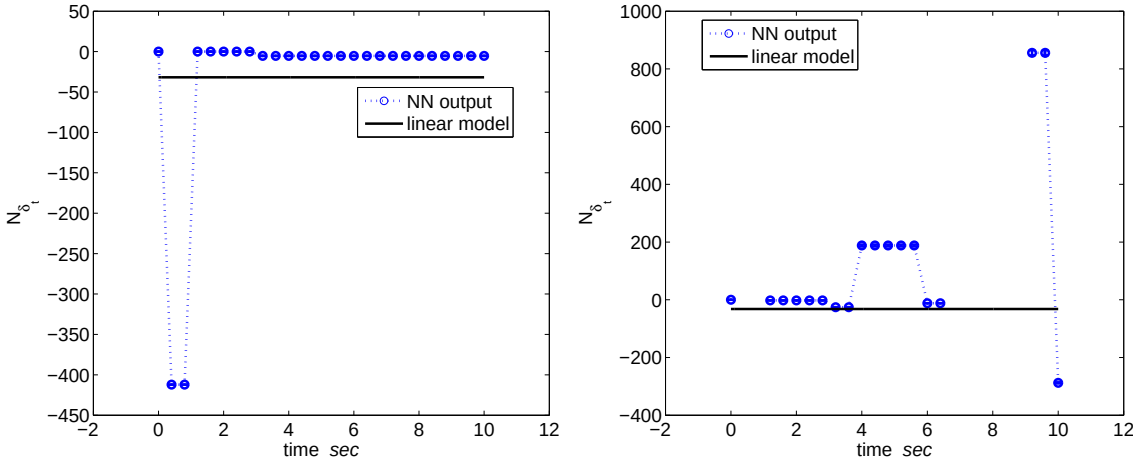

Figure 4.116: N_{δ_t} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.15: N_{δ_t} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(72)	0(0)	0(47)
stack - 100	6(113)	0(255)	0(0)	0(0)
stack - 150	0(219)	0(0)	0(0)	0(36)

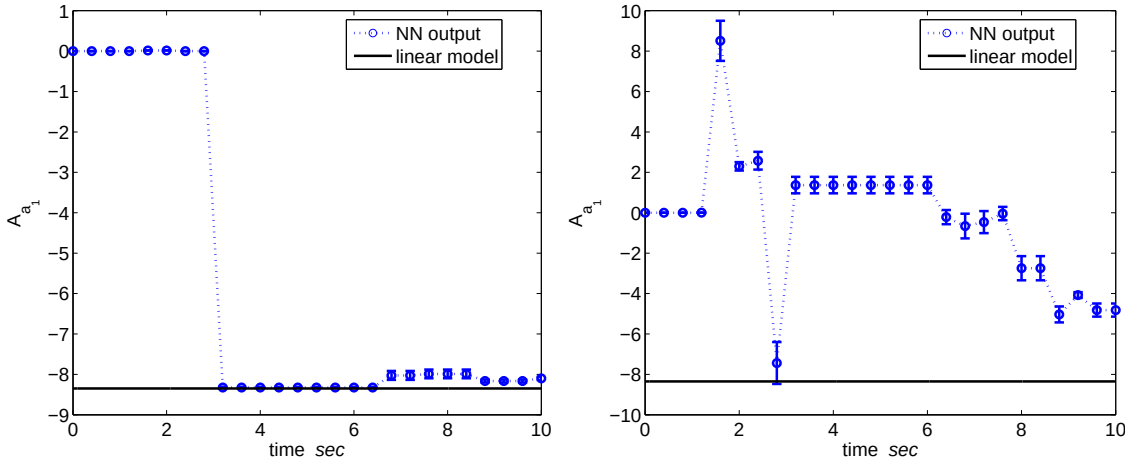

Figure 4.117: A_{a_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

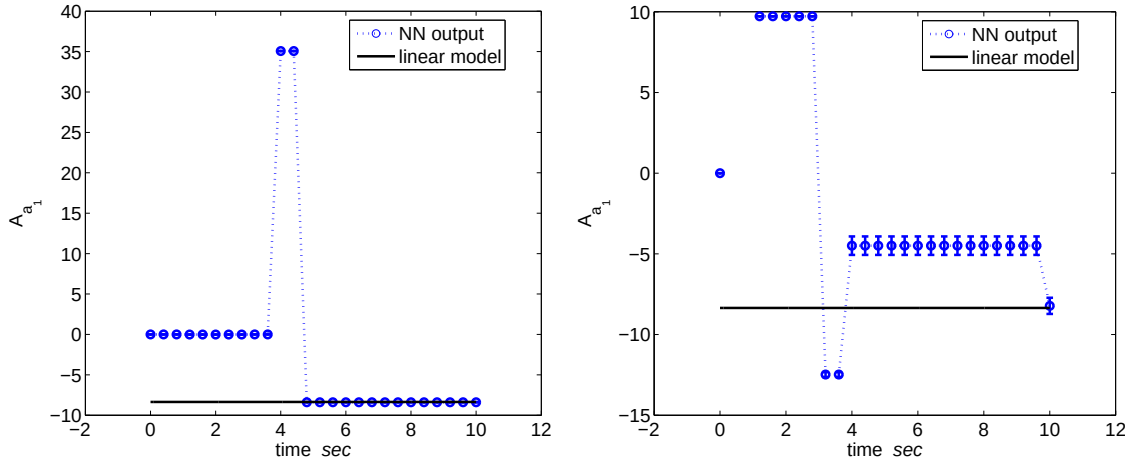

Figure 4.118: A_{a_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.16: A_{a_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	78(279)	77(90)	0(9)	0(76)
stack - 100	199(356)	0(284)	0(58)	197(242)
stack - 150	359(359)	265(265)	0(44)	16(16)

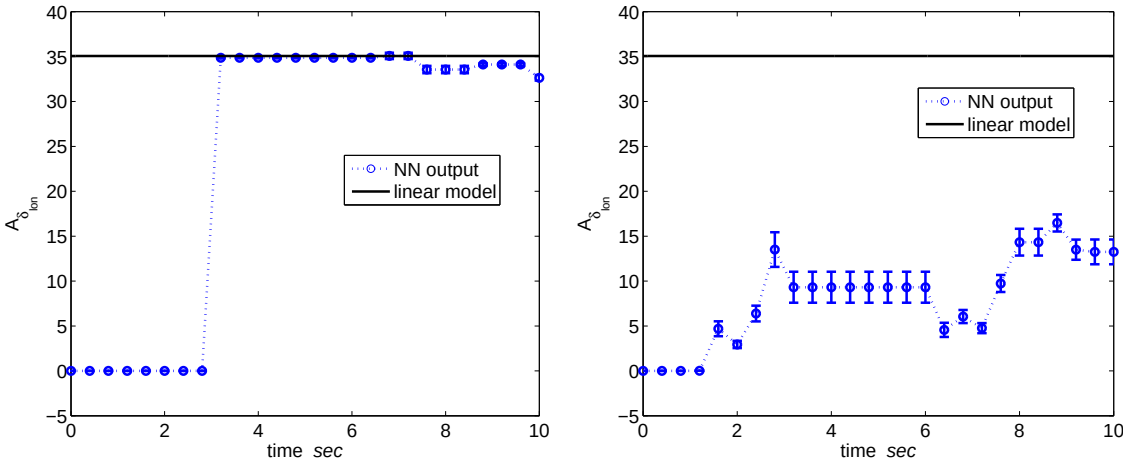

Figure 4.119: $A_{\delta_{lon}}$ hover flight DM online estimation: (1) ideal, (2) turbulence.

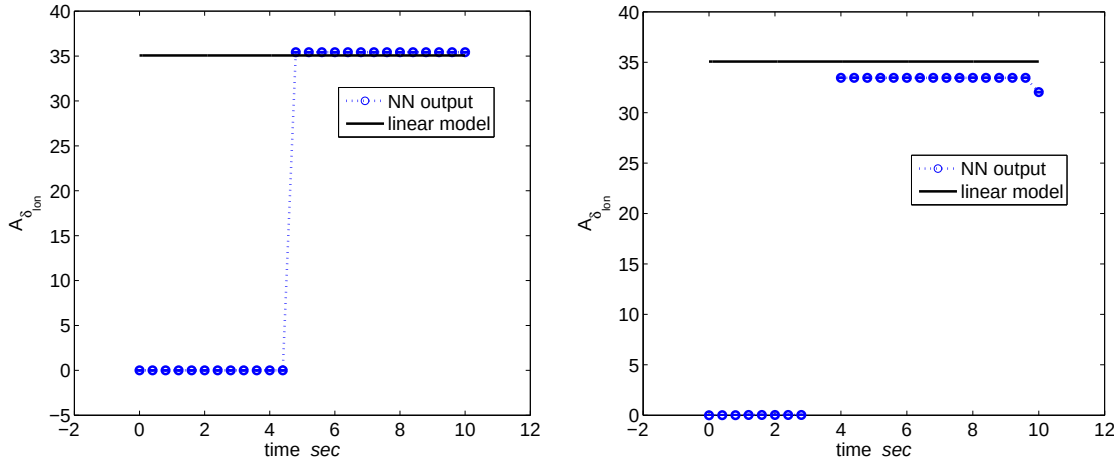

Figure 4.120: $A_{\delta_{lon}}$ hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.17: $A_{\delta_{lon}}$ hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(1)	0(0)	0(0)	0(51)
stack - 100	0(2)	15(74)	0(0)	0(0)
stack - 150	358(359)	265(265)	0(0)	299(315)

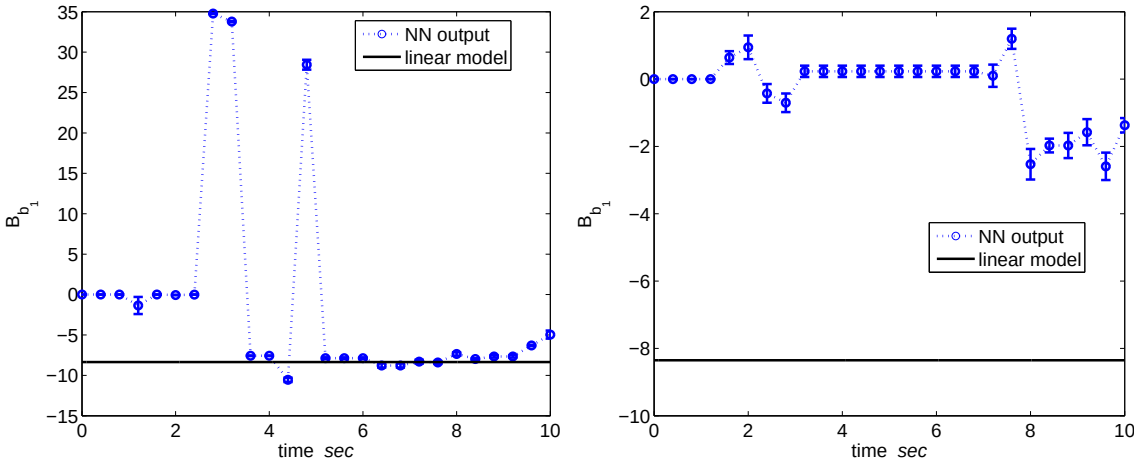

Figure 4.121: B_{b_1} hover flight DM online estimation: (1) ideal, (2) turbulence.

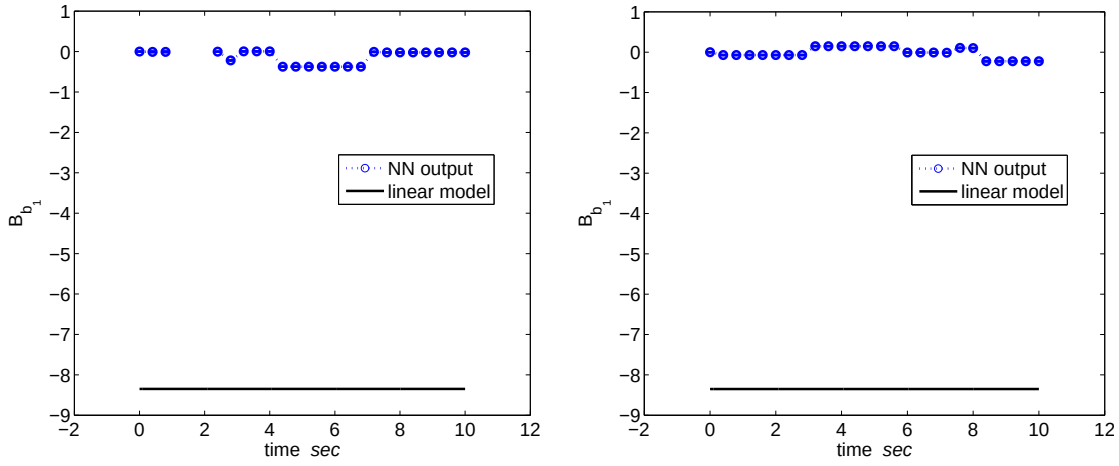

Figure 4.122: B_{b_1} hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.18: B_{b_1} hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	23(205)	0(0)	0(0)	0(0)
stack - 100	79(288)	0(0)	0(1)	0(0)
stack - 150	71(290)	0(0)	0(0)	0(0)

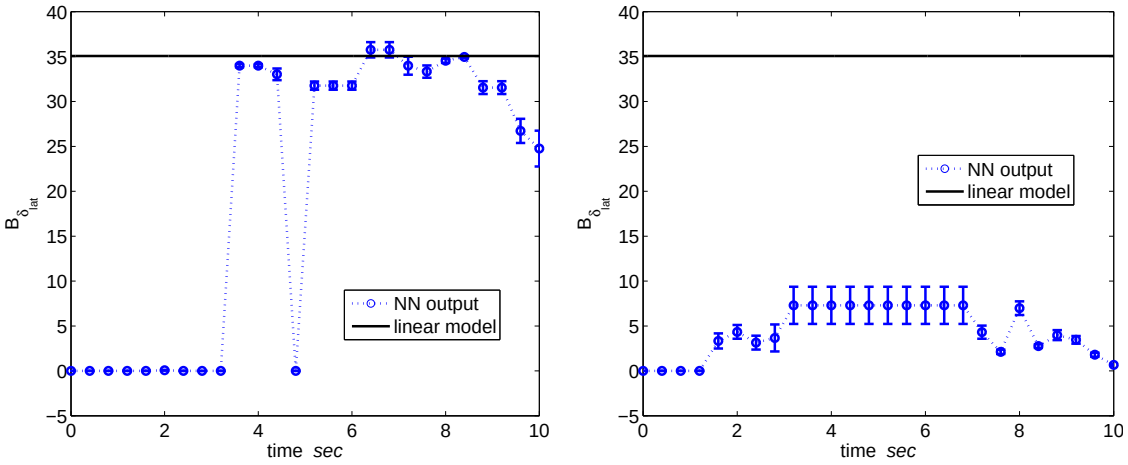

Figure 4.123: $B_{\delta_{lat}}$ hover flight DM online estimation: (1) ideal, (2) turbulence.

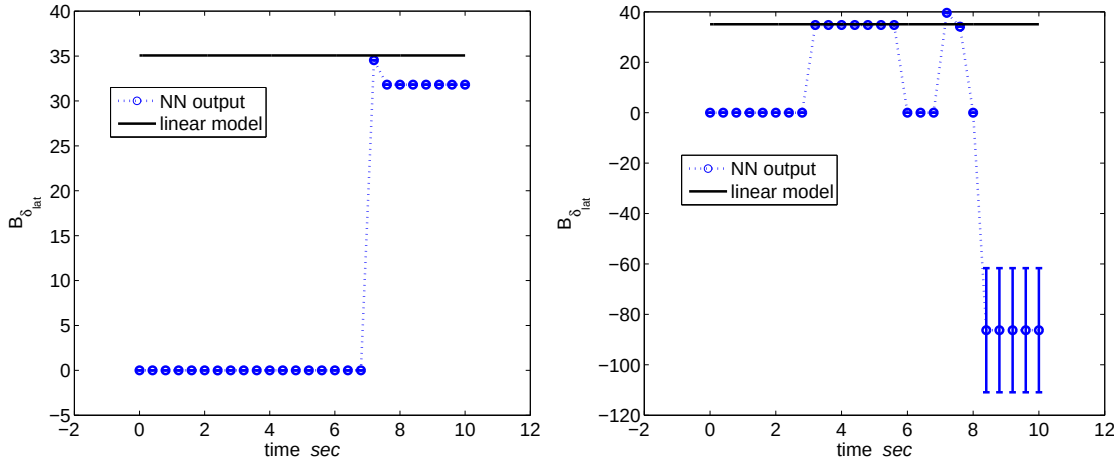

Figure 4.124: $B_{\delta_{lat}}$ hover flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.19: $B_{\delta_{lat}}$ hover flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	105(204)	12(64)	0(0)	0(0)
stack - 100	44(268)	113(114)	0(0)	0(36)
stack - 150	179(298)	7(140)	0(0)	158(180)

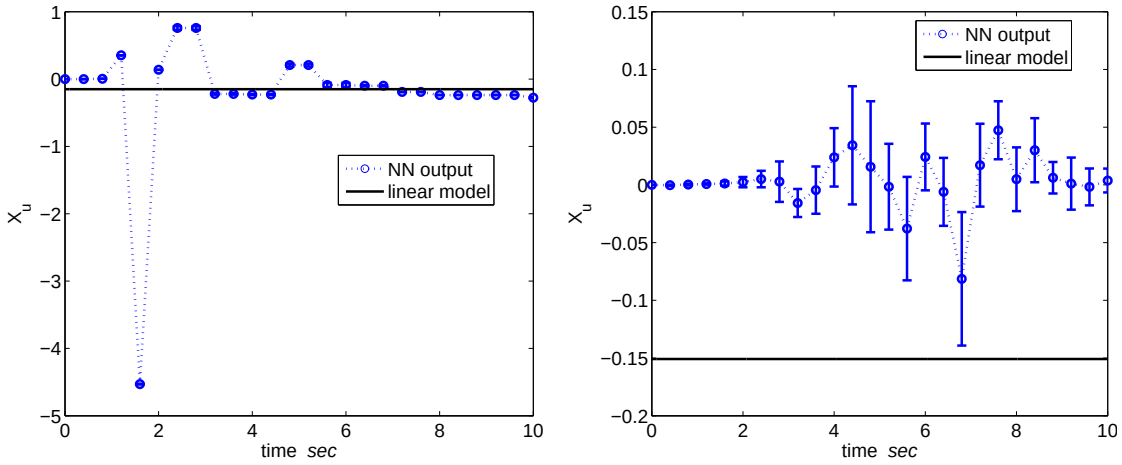

Figure 4.125: X_u forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

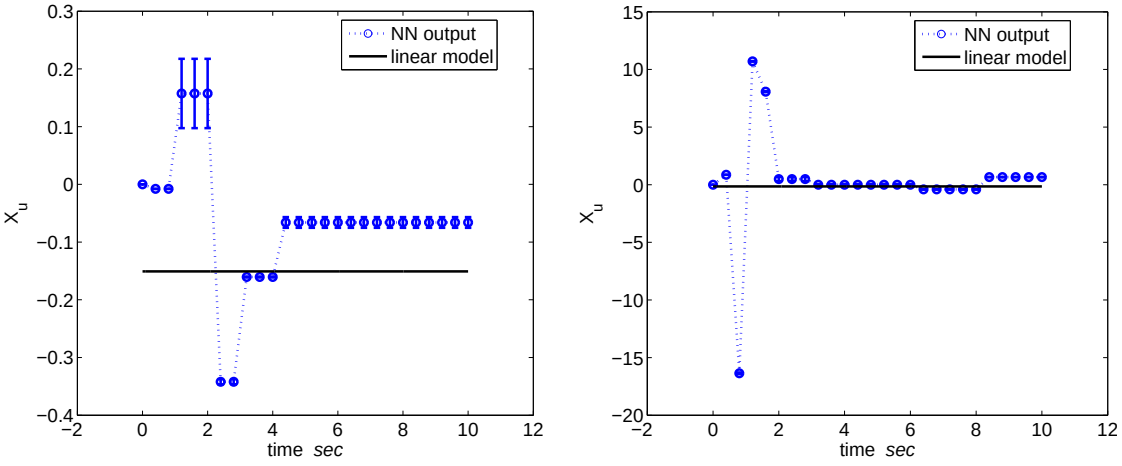

Figure 4.126: X_u forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.20: X_u forward 10m/s flight: No. of estimated values with 95(60)% confidence

	Ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(35)	0(9)	0(9)	0(0)
stack - 100	13(145)	0(109)	0(3)	0(63)
stack - 150	0(92)	0(65)	0(3)	0(0)

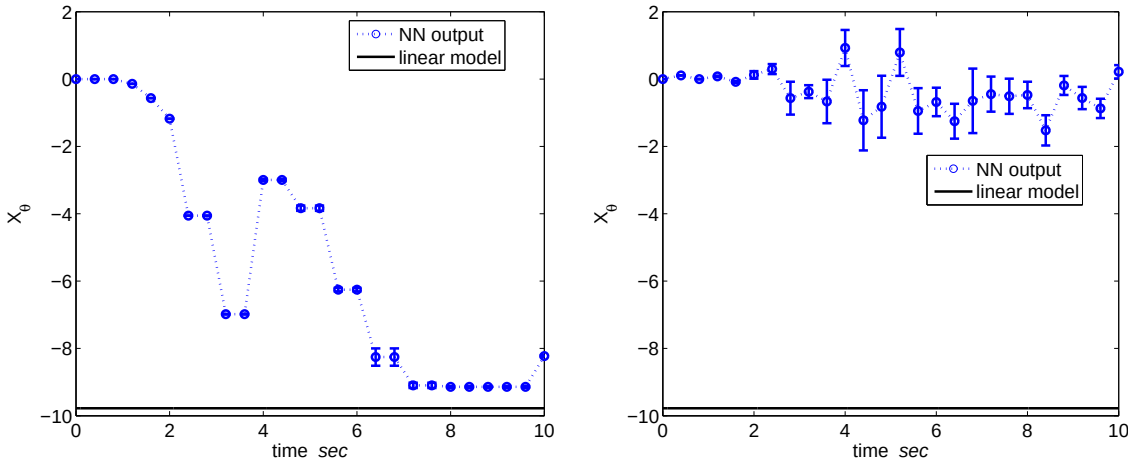

Figure 4.127: X_θ forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

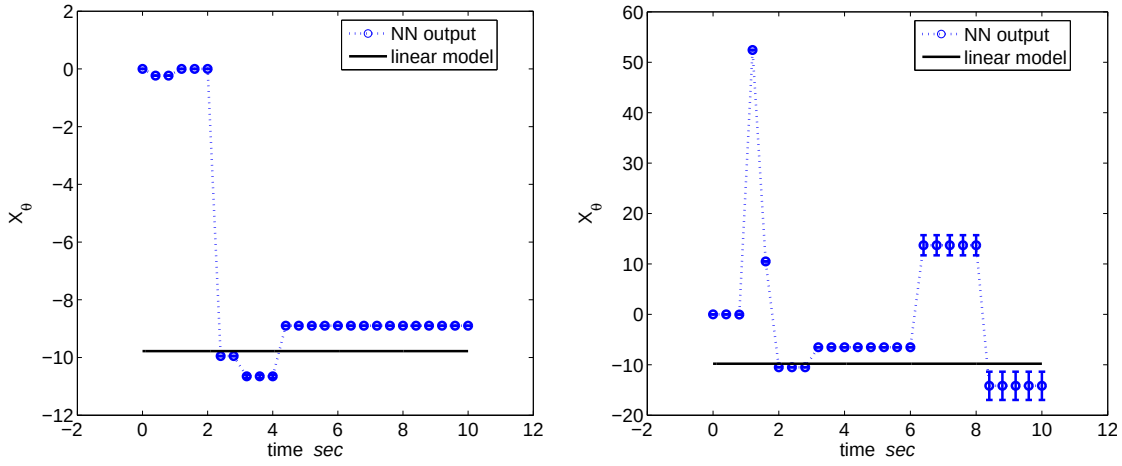

Figure 4.128: X_θ forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.21: X_θ forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(49)	44(102)	0(2)	0(0)
stack - 100	38(161)	110(233)	0(0)	0(1)
stack - 150	0(264)	42(393)	0(0)	0(218)

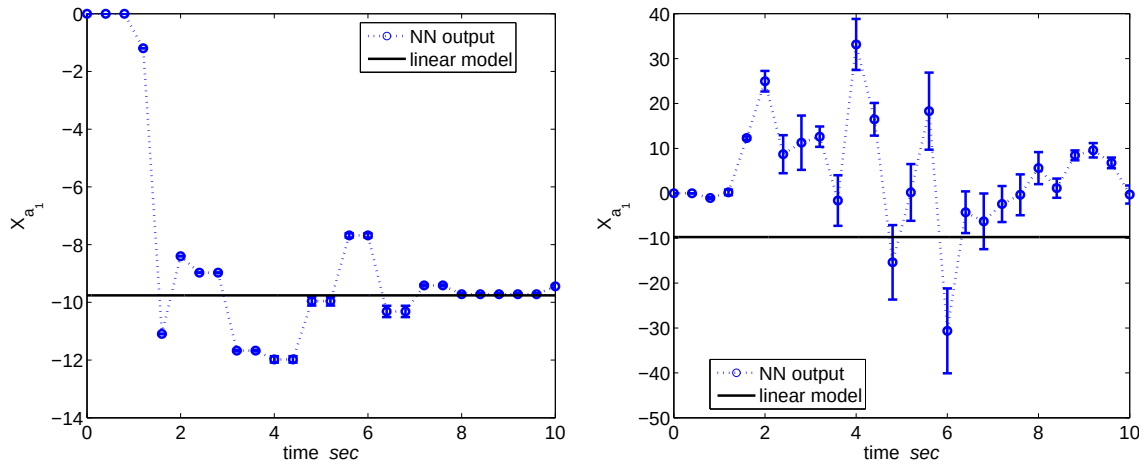

Figure 4.129: X_{a_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

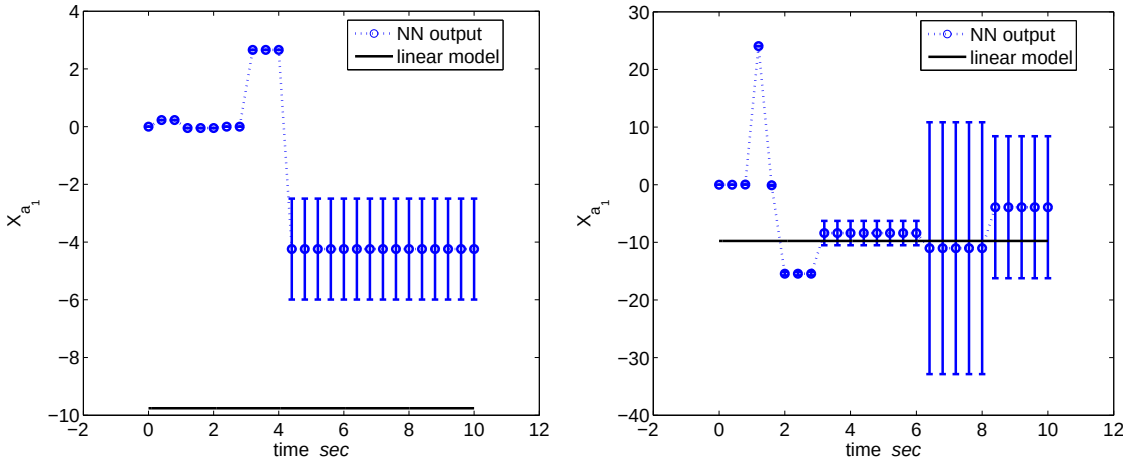

Figure 4.130: X_{a_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.22: X_{a_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	54(230)	22(205)	3(39)	0(5)
stack - 100	142(410)	133(257)	5(51)	1(63)
stack - 150	217(429)	0(0)	3(46)	0(271)

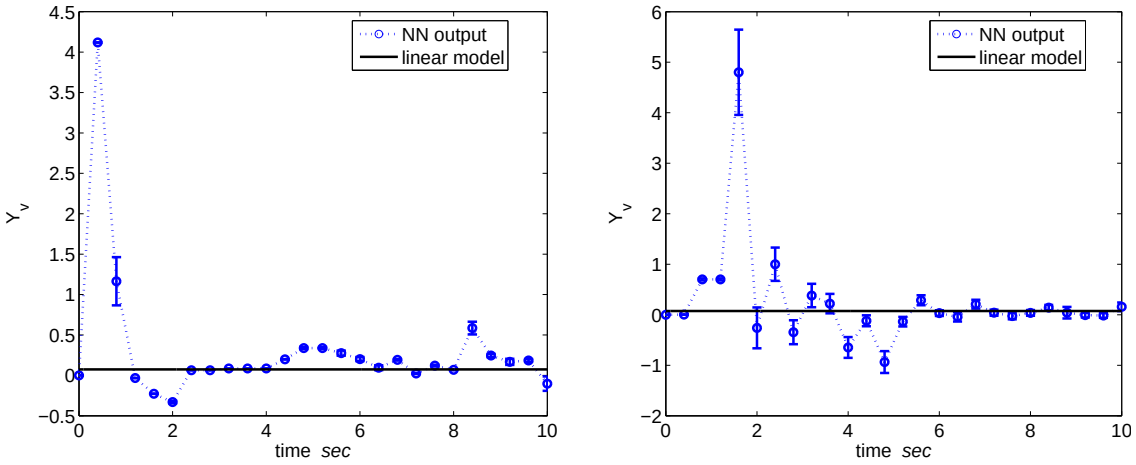

Figure 4.131: Y_v forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

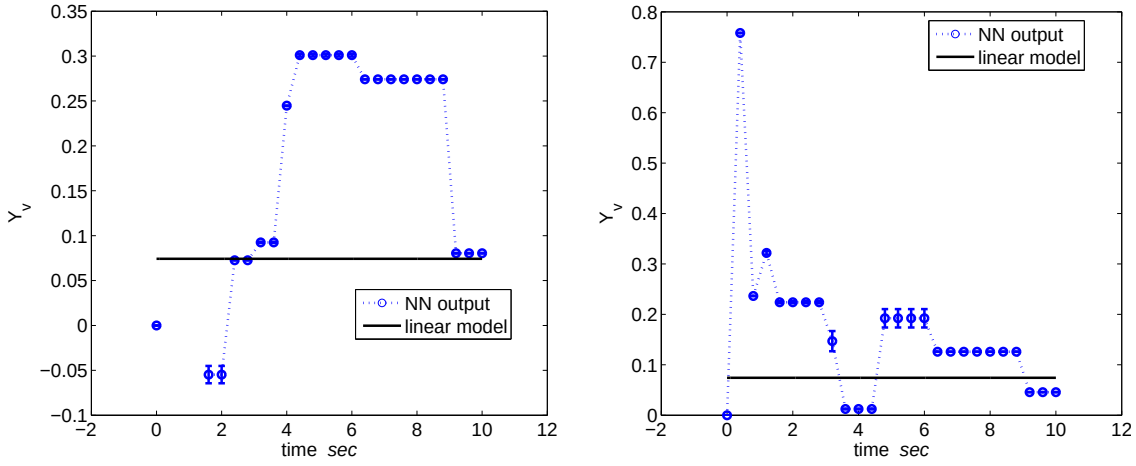

Figure 4.132: Y_v forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.23: Y_v forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(74)	27(98)	6(47)	0(106)
stack - 100	21(114)	0(0)	4(57)	0(0)
stack - 150	0(138)	27(129)	8(52)	0(46)

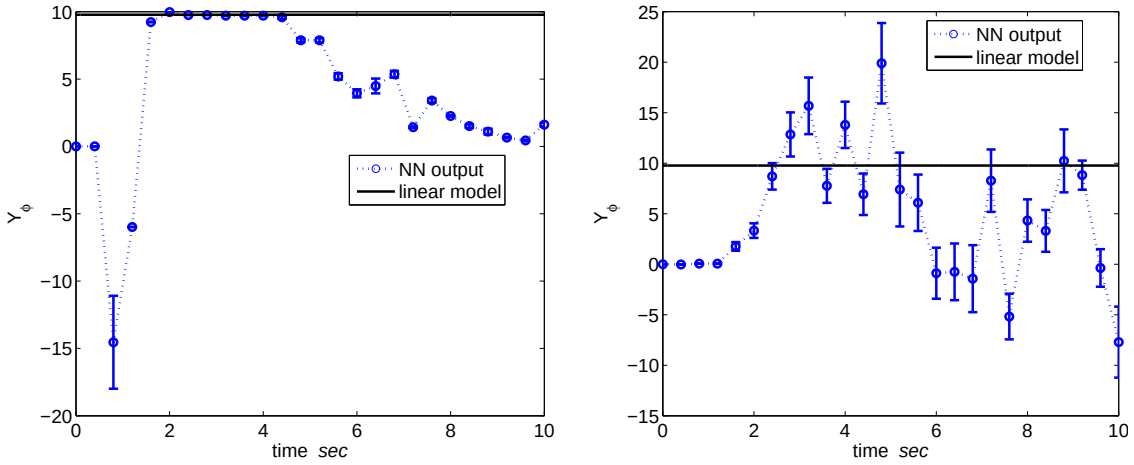

Figure 4.133: Y_ϕ forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

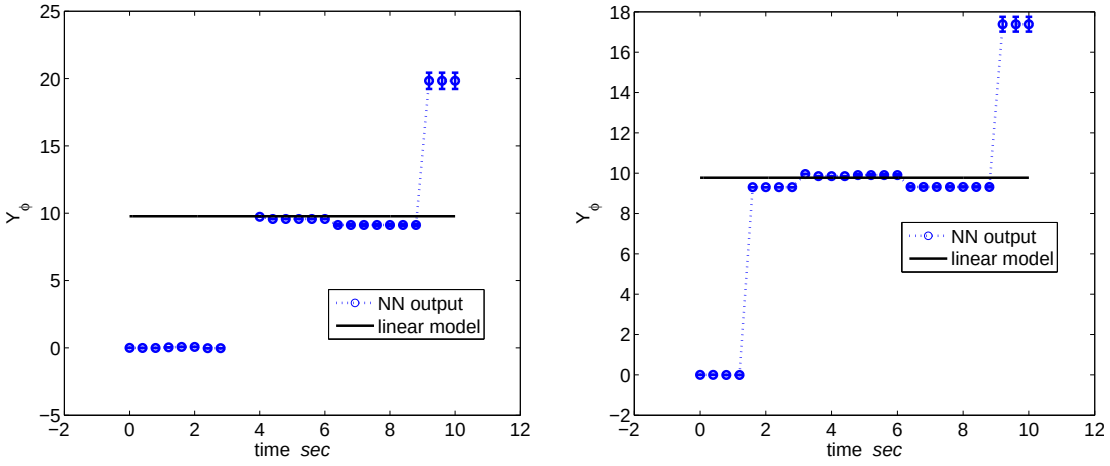

Figure 4.134: Y_ϕ forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.24: Y_ϕ forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	66(108)	0(137)	26(116)	0(127)
stack - 100	114(166)	89(141)	15(125)	140(206)
stack - 150	155(246)	112(263)	18(128)	387(387)

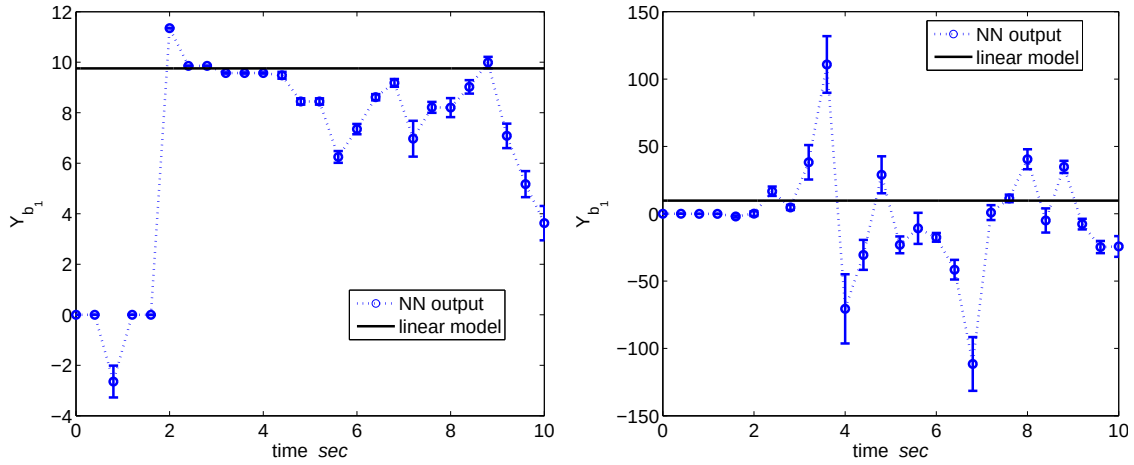

Figure 4.135: Y_{b_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

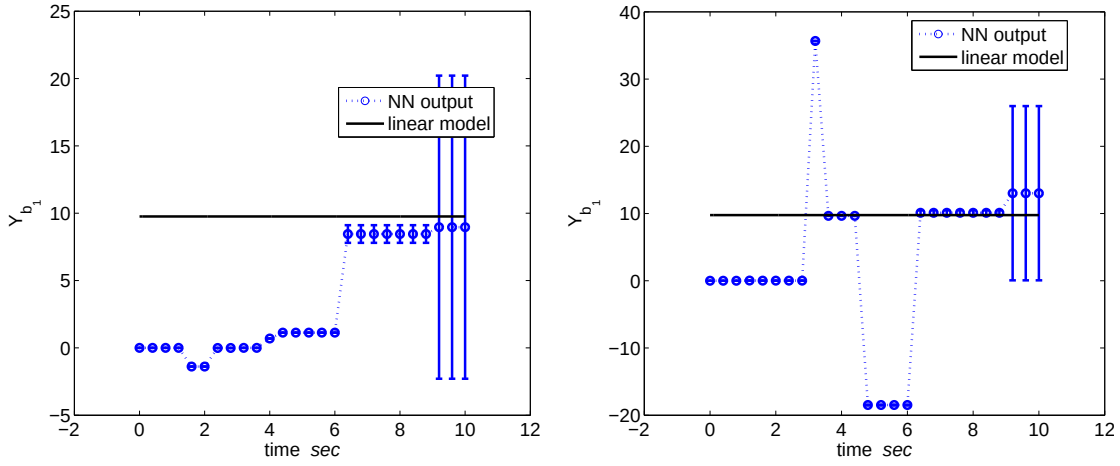

Figure 4.136: Y_{b_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.25: Y_{b_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	21(162)	0(43)	6(34)	0(51)
stack - 100	48(321)	0(0)	5(26)	31(99)
stack - 150	147(402)	0(193)	5(33)	203(249)

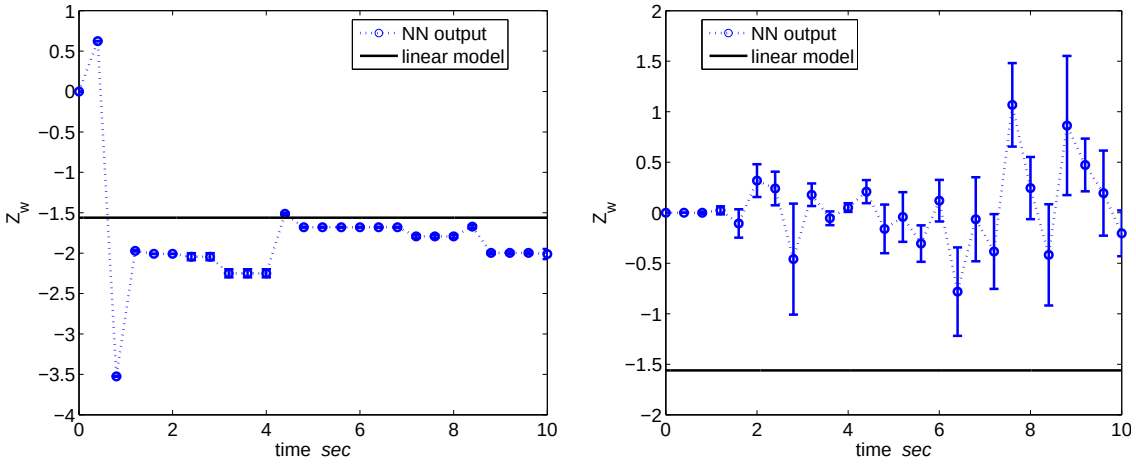

Figure 4.137: Z_w forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

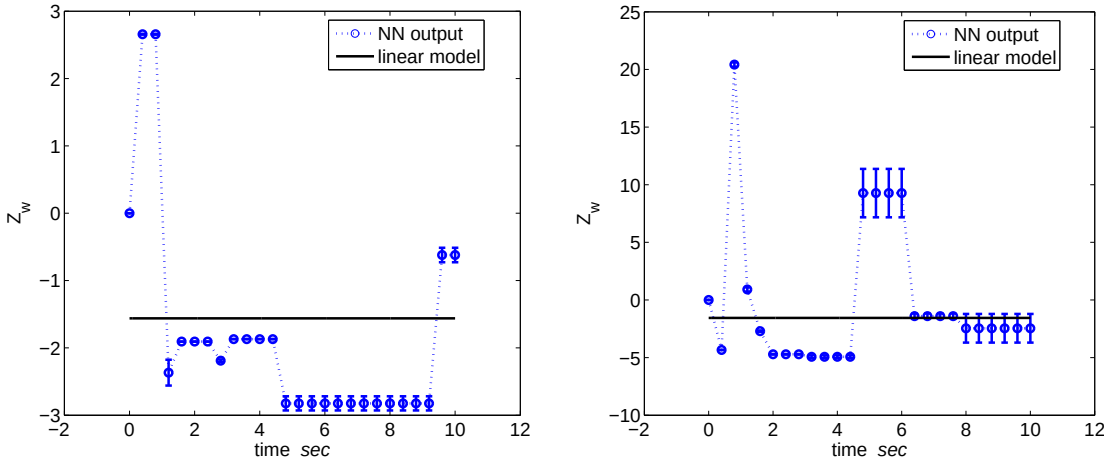

Figure 4.138: Z_w forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.26: Z_w forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	3(386)	128(242)	0(9)	0(0)
stack - 100	32(410)	12(366)	1(5)	1(12)
stack - 150	22(385)	0(133)	0(0)	3(109)

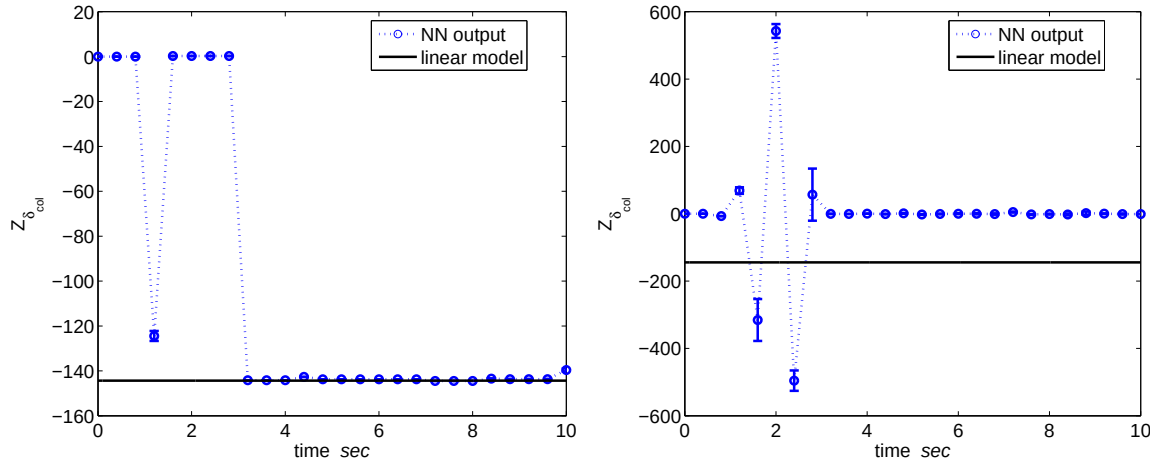

Figure 4.139: $Z_{\delta_{col}}$ forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

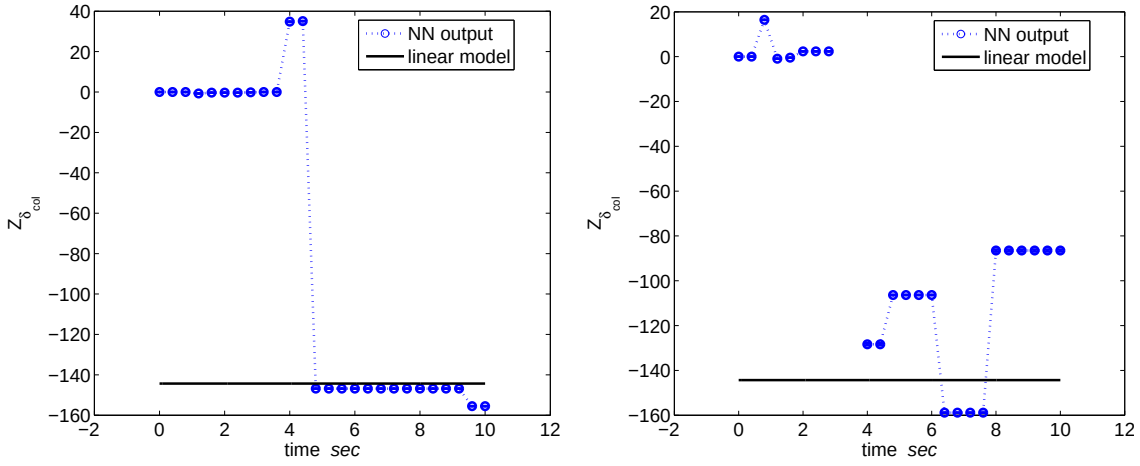

Figure 4.140: $Z_{\delta_{col}}$ forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.27: $Z_{\delta_{col}}$ forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	335(335)	10(65)	0(0)	12(12)
stack - 100	399(403)	184(184)	1(10)	0(87)
stack - 150	353(365)	245(280)	0(8)	1(217)

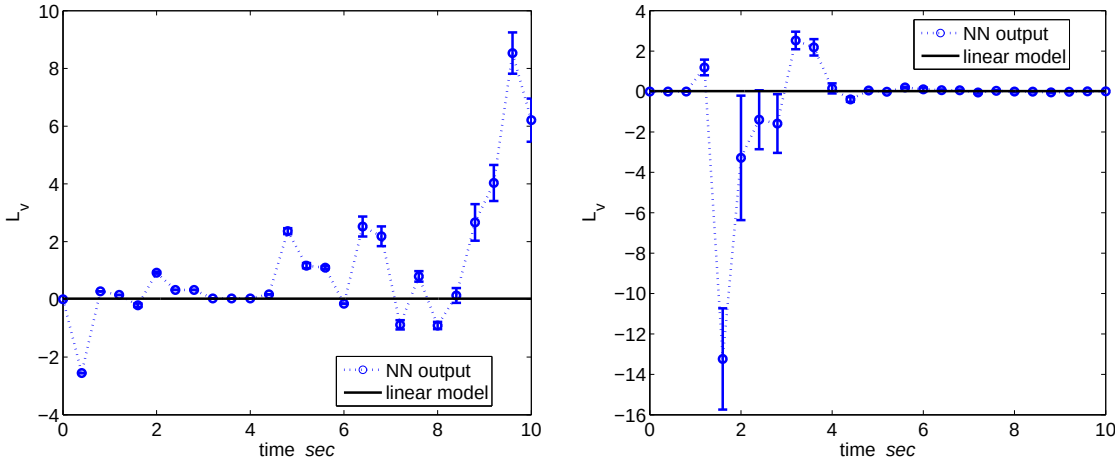

Figure 4.141: L_v forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

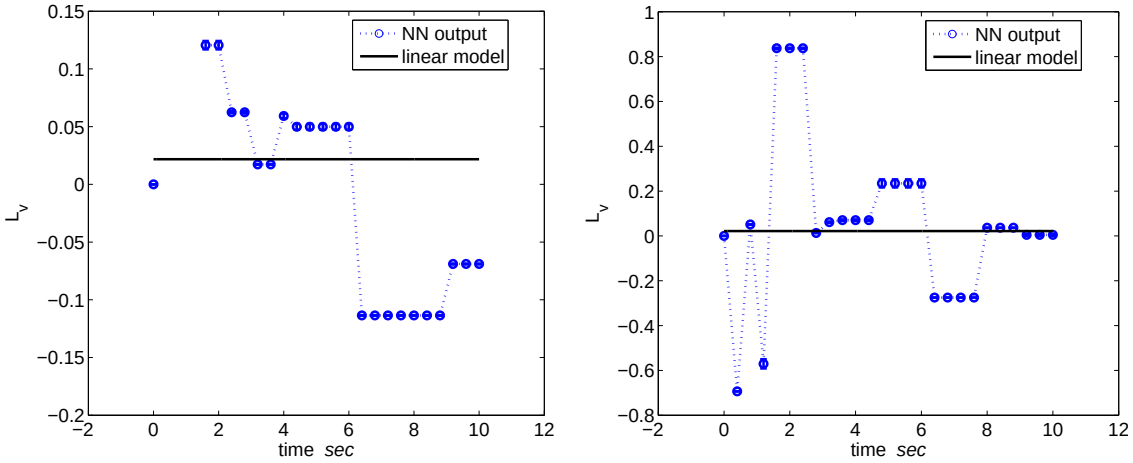

Figure 4.142: L_v forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.28: L_v forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(16)	0(9)	5(22)	0(0)
stack - 100	0(0)	0(0)	3(35)	0(0)
stack - 150	0(55)	4(50)	2(26)	0(11)

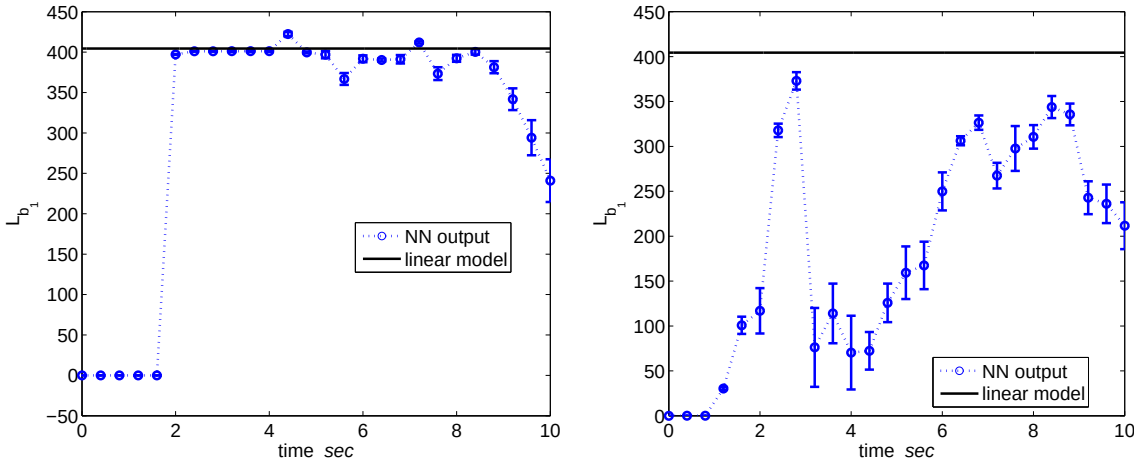

Figure 4.143: L_{b_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

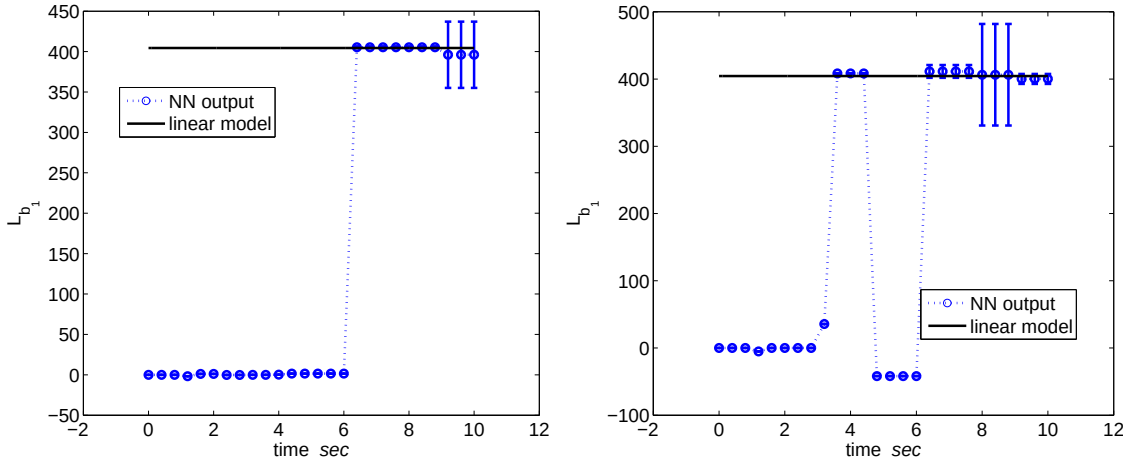

Figure 4.144: L_{b_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.29: L_{b_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	56(260)	52(80)	9(117)	0(37)
stack - 100	161(365)	133(181)	1(210)	135(195)
stack - 150	294(413)	202(202)	0(200)	249(249)

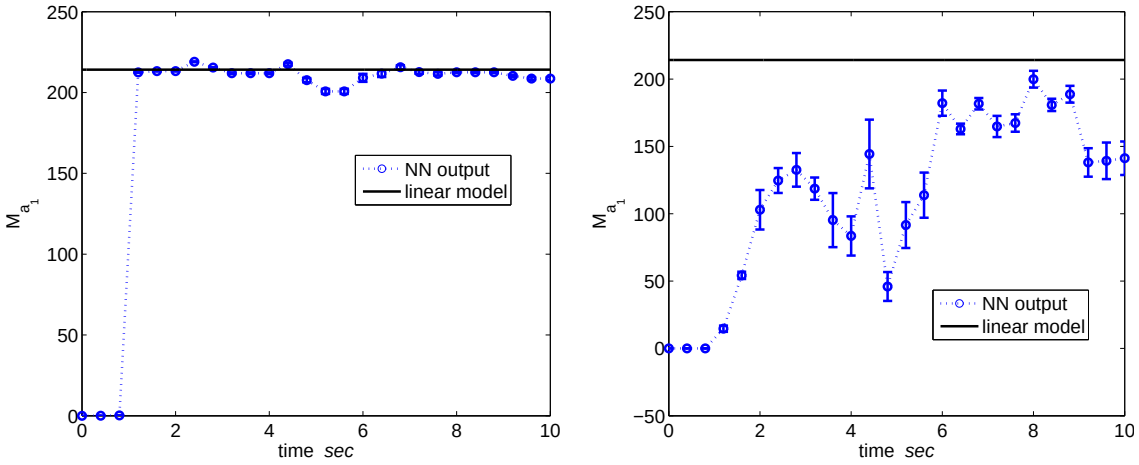

Figure 4.145: M_{a_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

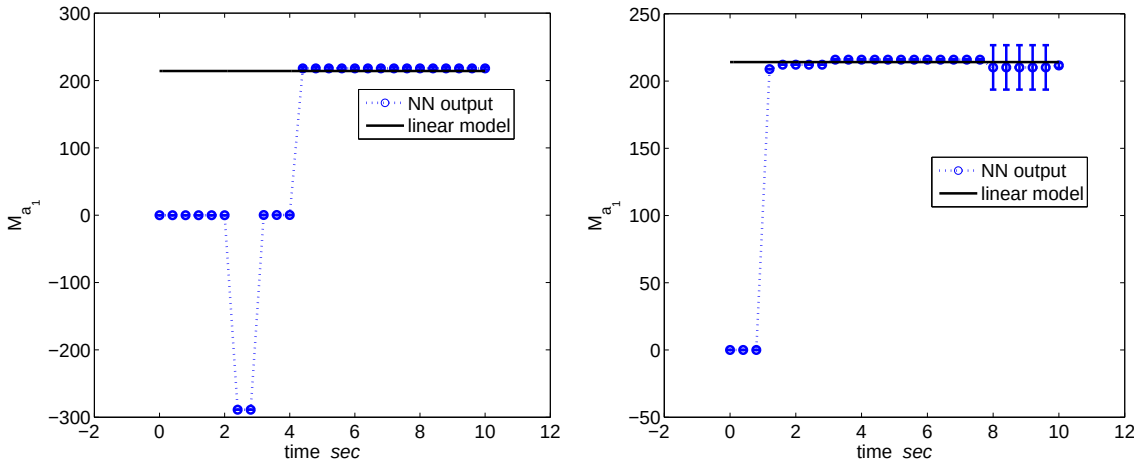

Figure 4.146: M_{a_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.30: M_{a_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	294(311)	43(54)	18(211)	35(204)
stack - 100	400(444)	282(295)	13(259)	165(445)
stack - 150	405(446)	286(286)	5(227)	443(445)

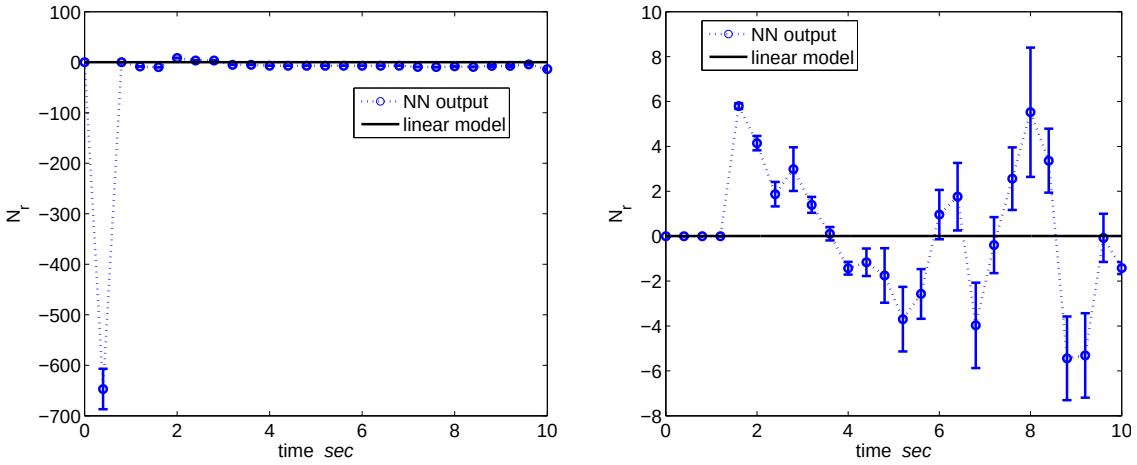

Figure 4.147: N_r forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

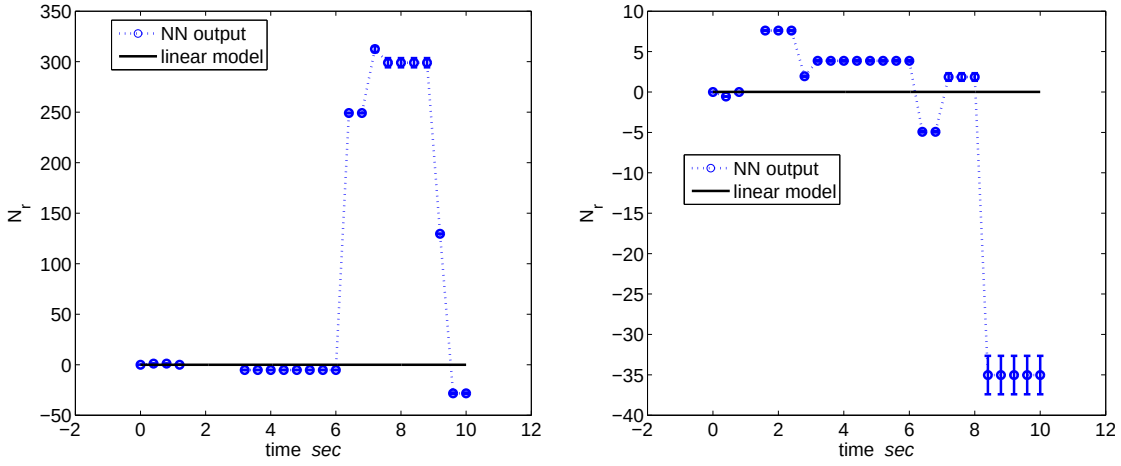

Figure 4.148: N_r forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.31: N_r forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(0)	0(0)	0(1)
stack - 100	0(0)	0(1)	0(1)	0(1)
stack - 150	0(6)	0(12)	0(1)	2(7)

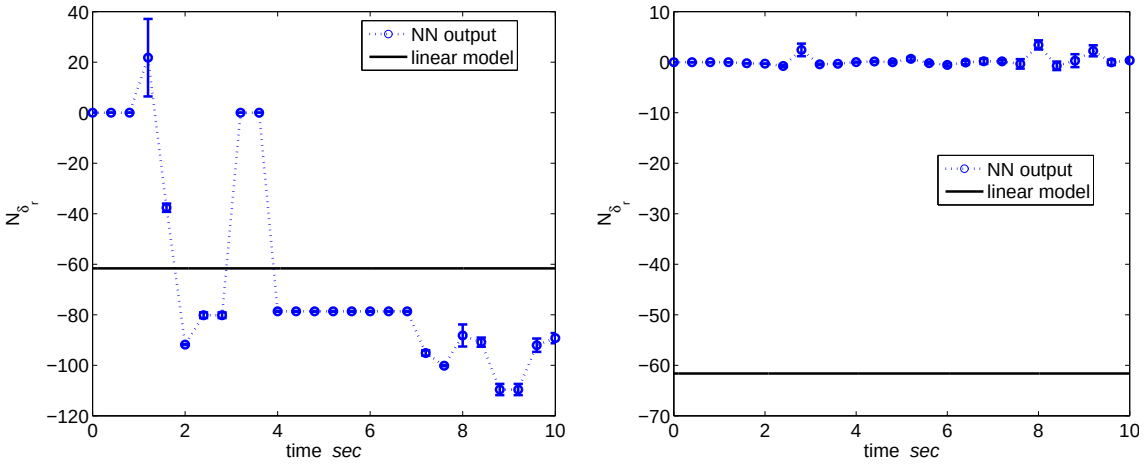

Figure 4.149: N_{δ_r} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

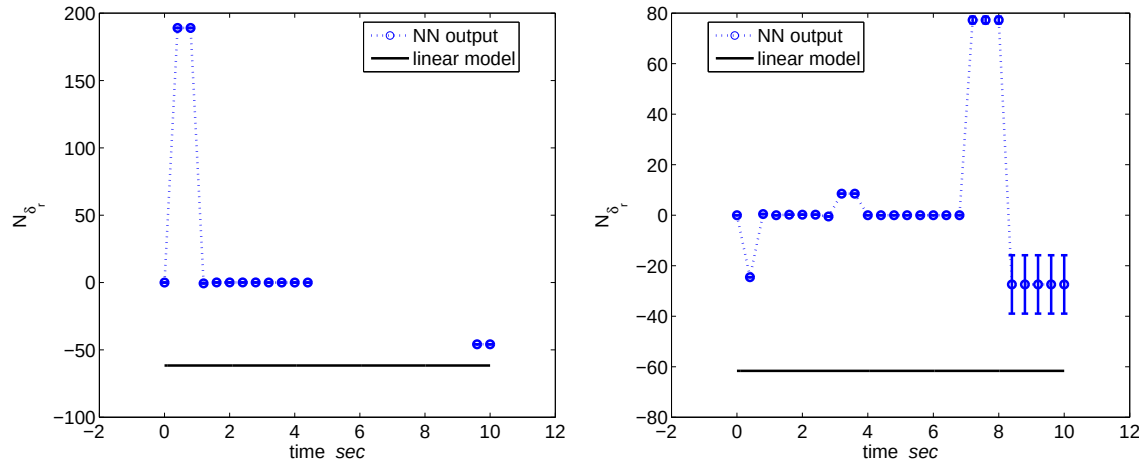

Figure 4.150: N_{δ_r} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.32: N_{δ_r} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(34)	0(1)	0(0)	47(49)
stack - 100	0(86)	0(0)	0(0)	0(0)
stack - 150	0(213)	0(26)	0(0)	1(14)

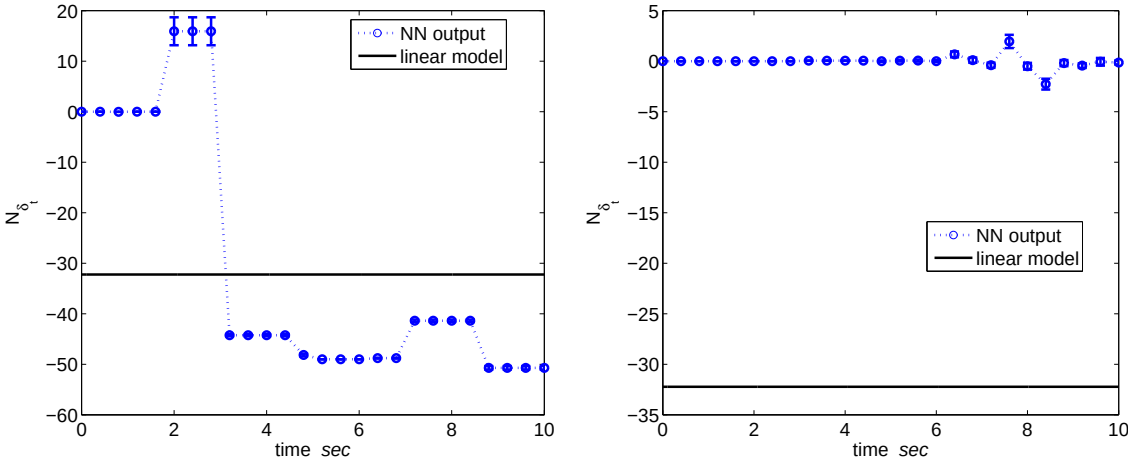

Figure 4.151: N_{δ_t} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

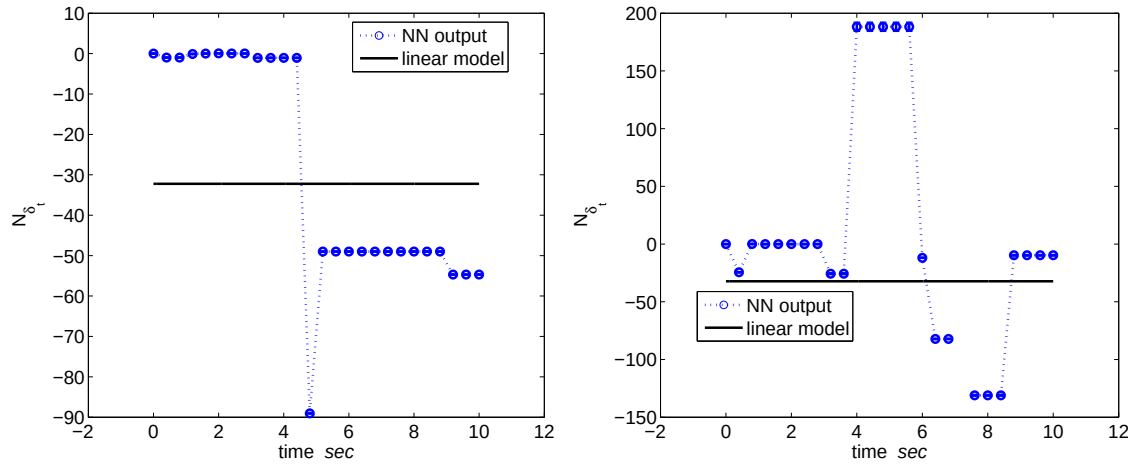

Figure 4.152: N_{δ_t} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.33: N_{δ_t} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(12)	2(26)	0(0)	0(10)
stack - 100	0(89)	0(202)	0(0)	0(4)
stack - 150	0(165)	0(0)	0(0)	0(38)

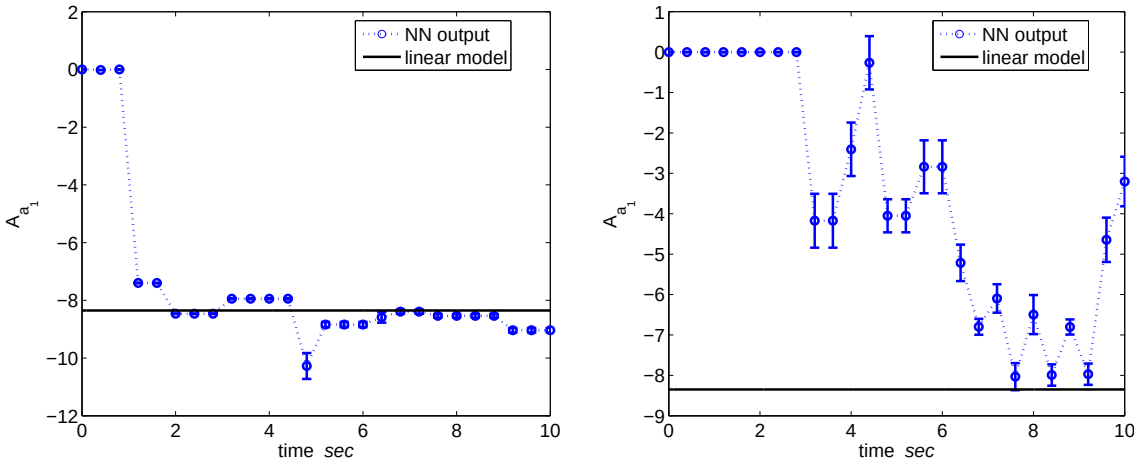

Figure 4.153: A_{a_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

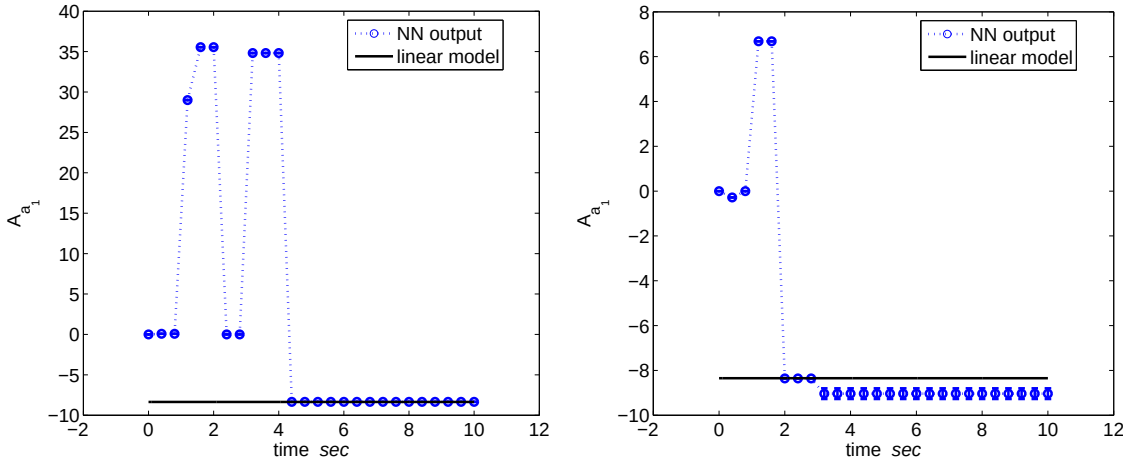

Figure 4.154: A_{a_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.34: A_{a_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	185(257)	0(17)	49(166)	1(128)
stack - 100	359(428)	148(148)	60(254)	194(234)
stack - 150	289(441)	286(286)	53(174)	53(405)

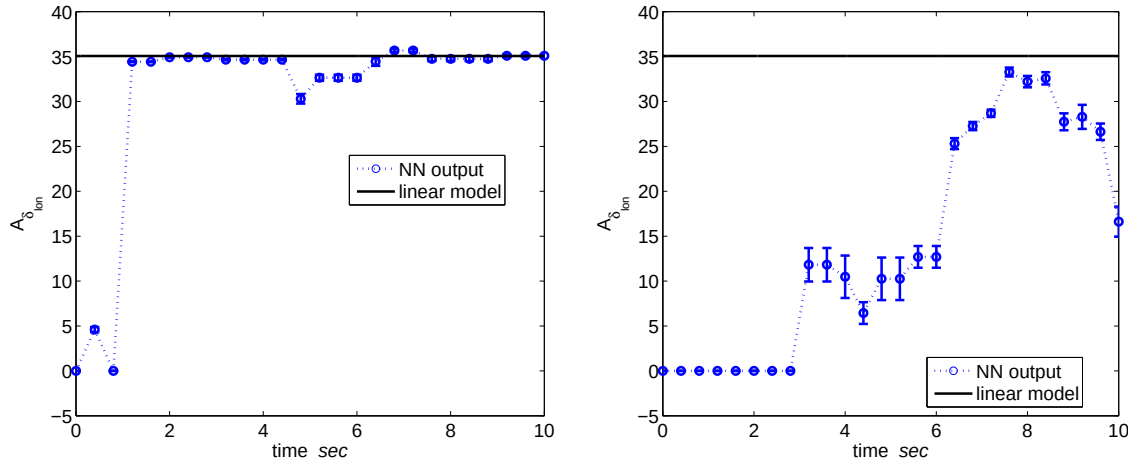

Figure 4.155: $A_{\delta_{lon}}$ forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

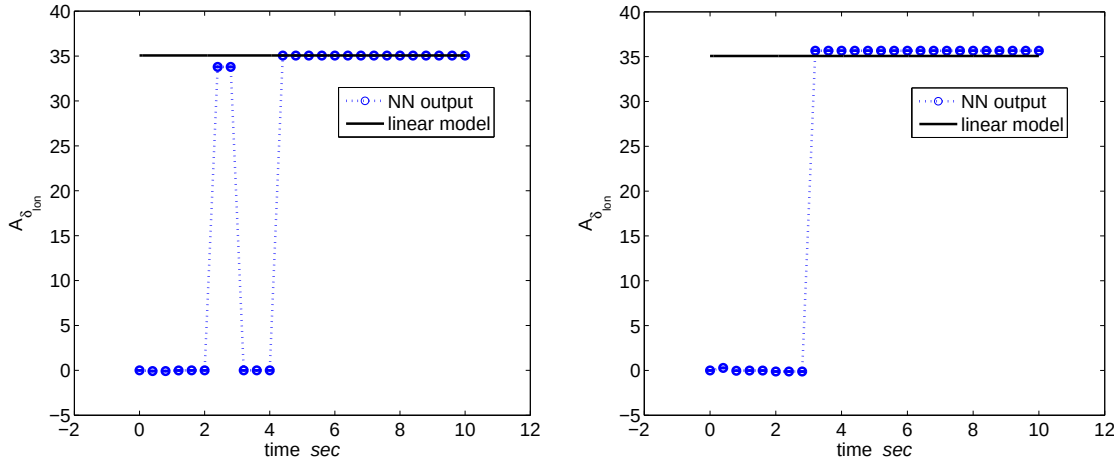

Figure 4.156: $A_{\delta_{lon}}$ forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.35: $A_{\delta_{lon}}$ forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(3)	0(0)	0(5)
stack - 100	0(0)	37(38)	0(0)	0(0)
stack - 150	369(446)	328(328)	4(189)	351(351)

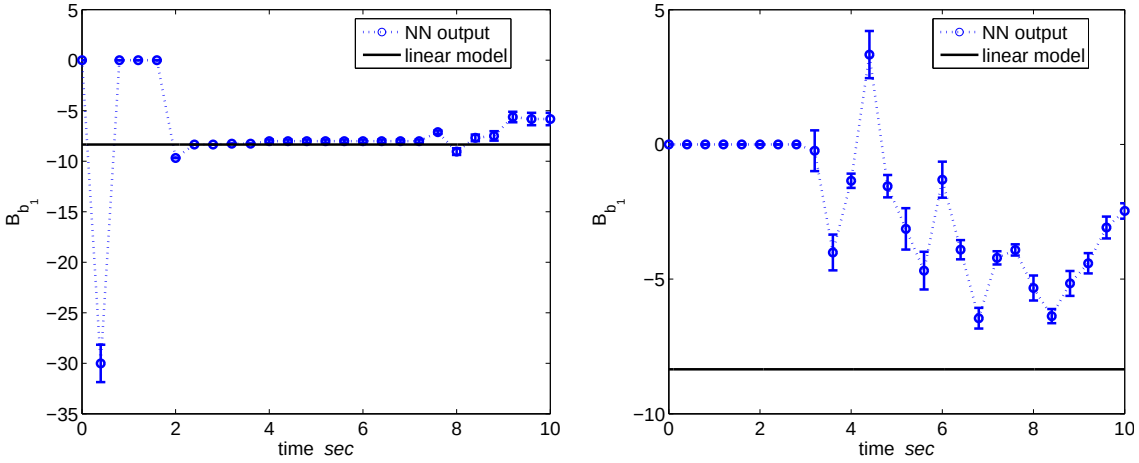

Figure 4.157: B_{b_1} forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

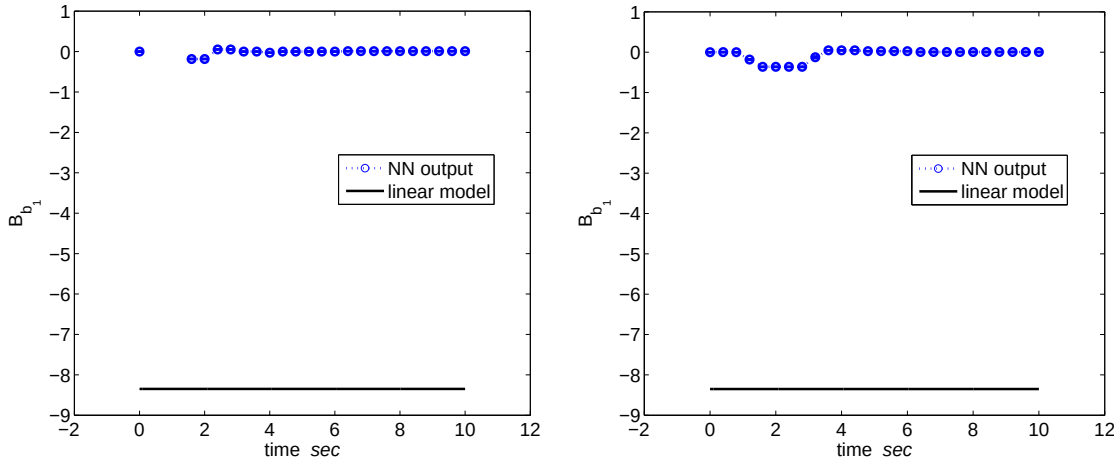

Figure 4.158: B_{b_1} forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.36: B_{b_1} forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	64(238)	0(0)	1(85)	0(0)
stack - 100	152(325)	0(0)	1(120)	0(0)
stack - 150	254(417)	0(0)	0(85)	0(0)

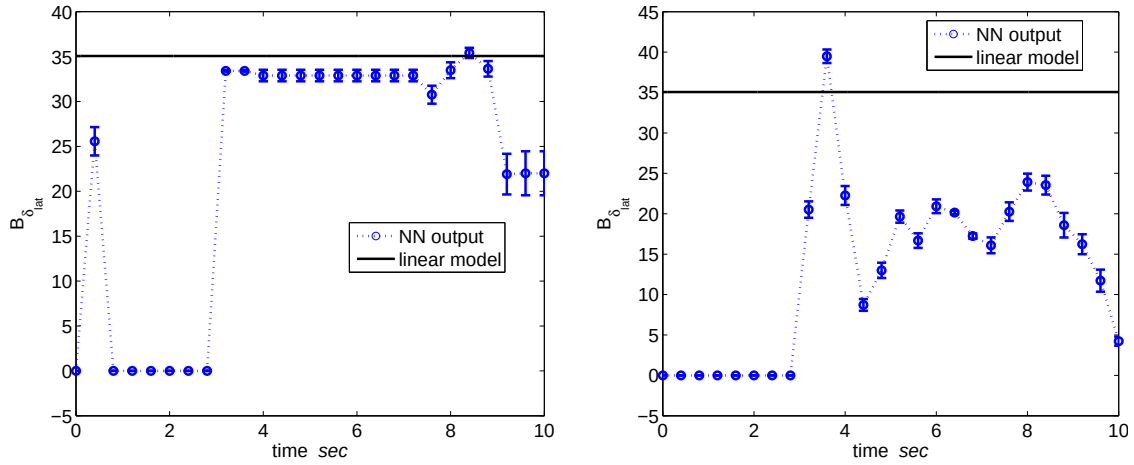

Figure 4.159: $B_{\delta_{lat}}$ forward 10m/s flight DM online estimation: (1) ideal, (2) turbulence.

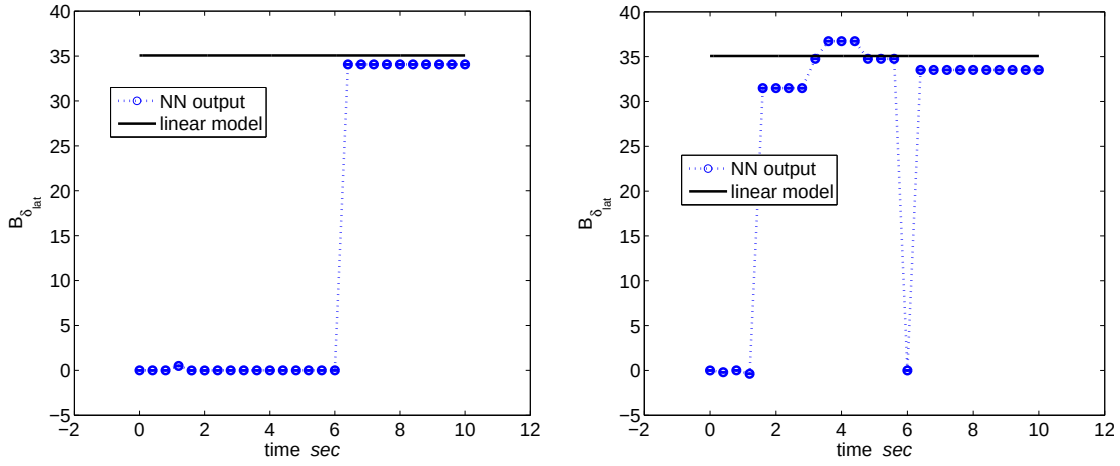

Figure 4.160: $B_{\delta_{lat}}$ forward 10m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.37: $B_{\delta_{lat}}$ forward 10m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	22(228)	0(64)	0(0)	0(0)
stack - 100	81(329)	37(127)	55(76)	140(176)
stack - 150	95(379)	206(206)	0(73)	339(421)

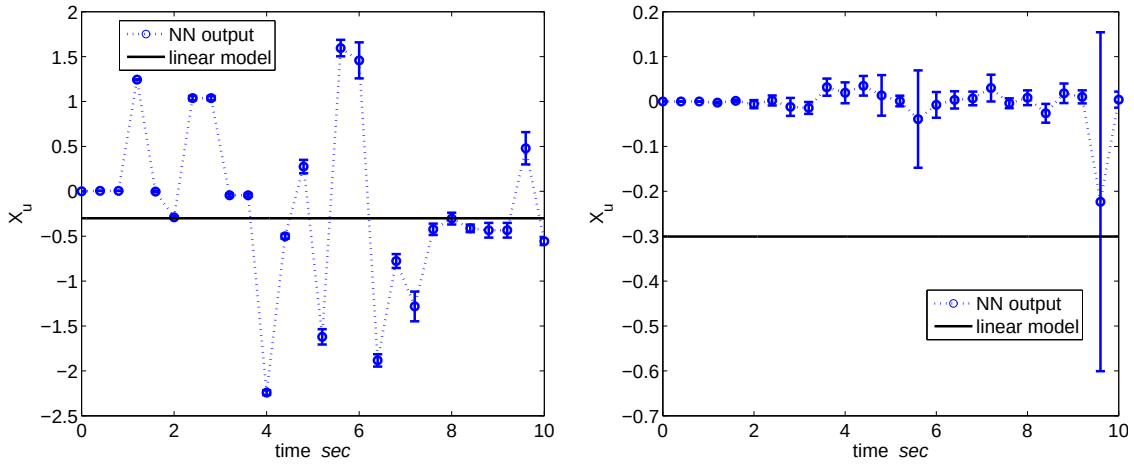

Figure 4.161: X_u forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

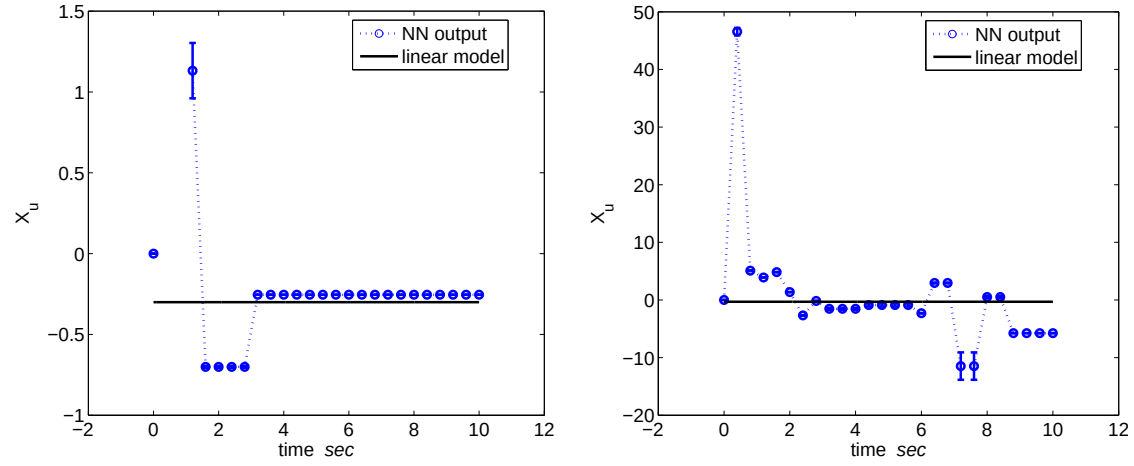

Figure 4.162: X_u forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.38: X_u forward 20m/s flight: No. of estimated values with 95(60)% confidence

	Ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	21(91)	0(12)	0(1)	0(0)
stack - 100	0(58)	0(32)	0(0)	0(0)
stack - 150	47(84)	0(351)	0(2)	2(5)

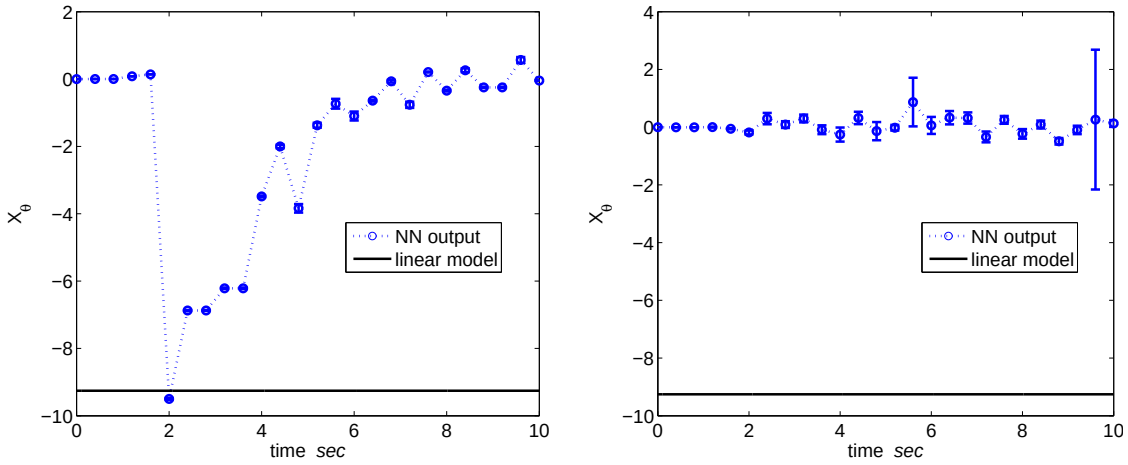

Figure 4.163: X_θ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

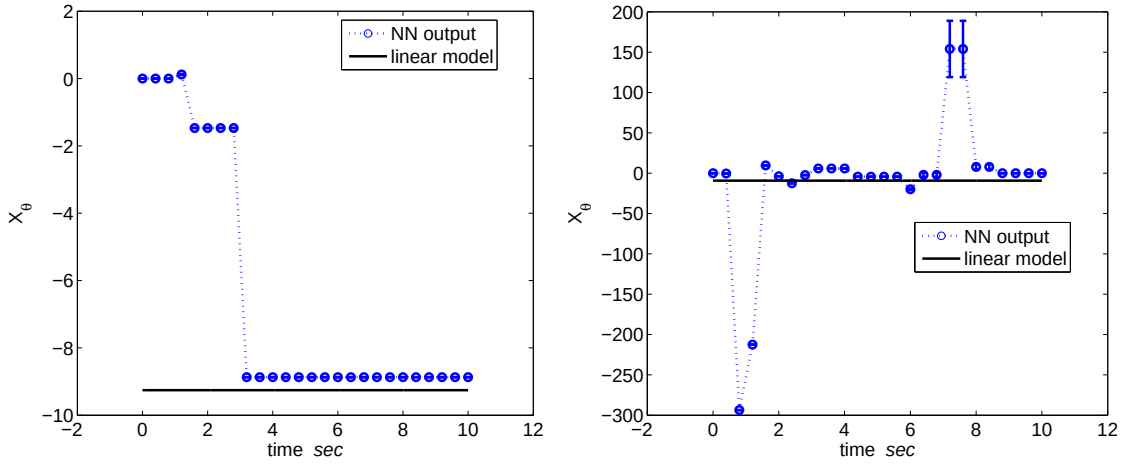

Figure 4.164: X_θ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.39: X_θ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	92(100)	1(1)	0(0)
stack - 100	0(71)	32(152)	0(0)	1(106)
stack - 150	24(97)	351(355)	0(0)	1(12)

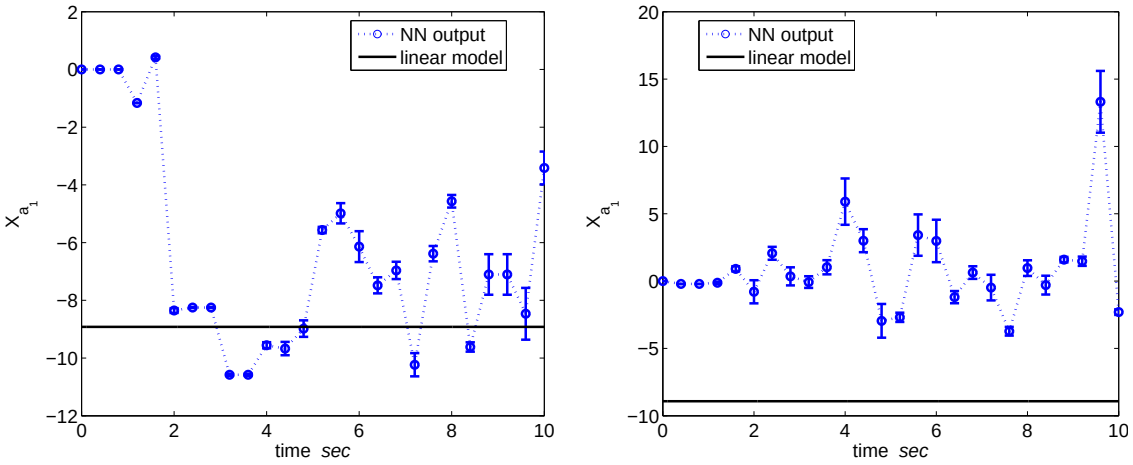

Figure 4.165: X_{a_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

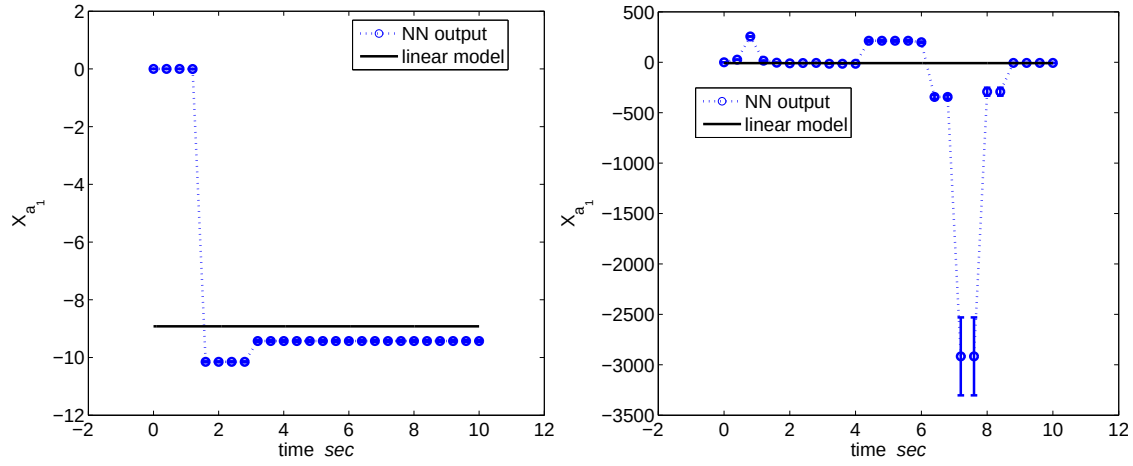

Figure 4.166: X_{a_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.40: X_{a_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	16(58)	35(60)	1(11)	0(0)
stack - 100	17(155)	60(126)	0(2)	2(69)
stack - 150	12(332)	0(430)	0(9)	1(132)

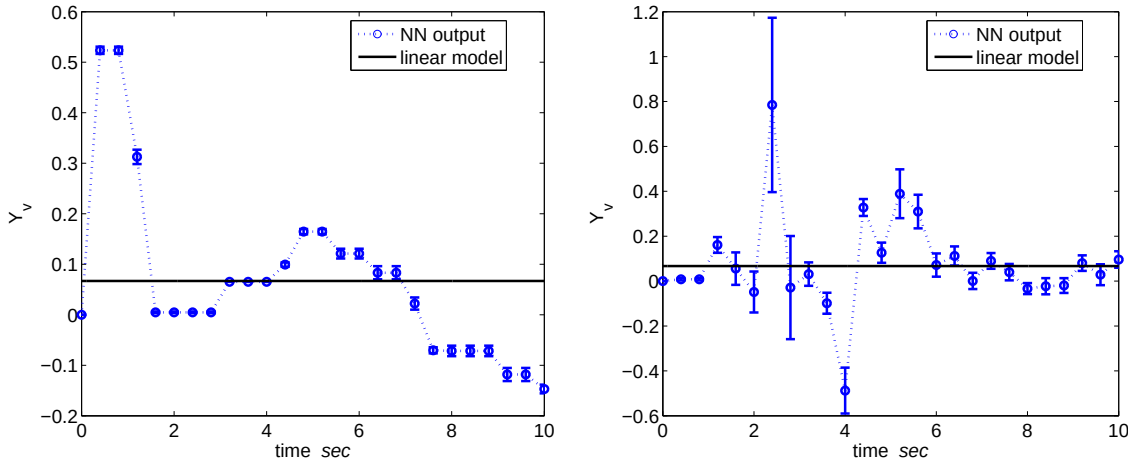

Figure 4.167: Y_v forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

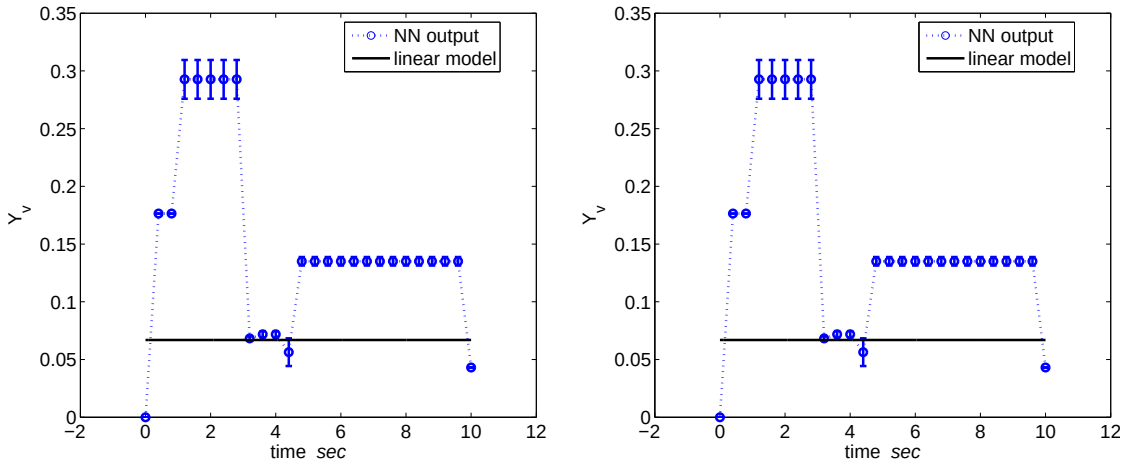

Figure 4.168: Y_v forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.41: Y_v forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	25(89)	0(7)	8(48)	0(0)
stack - 100	10(123)	0(62)	7(48)	0(0)
stack - 150	59(92)	20(96)	5(72)	0(4)

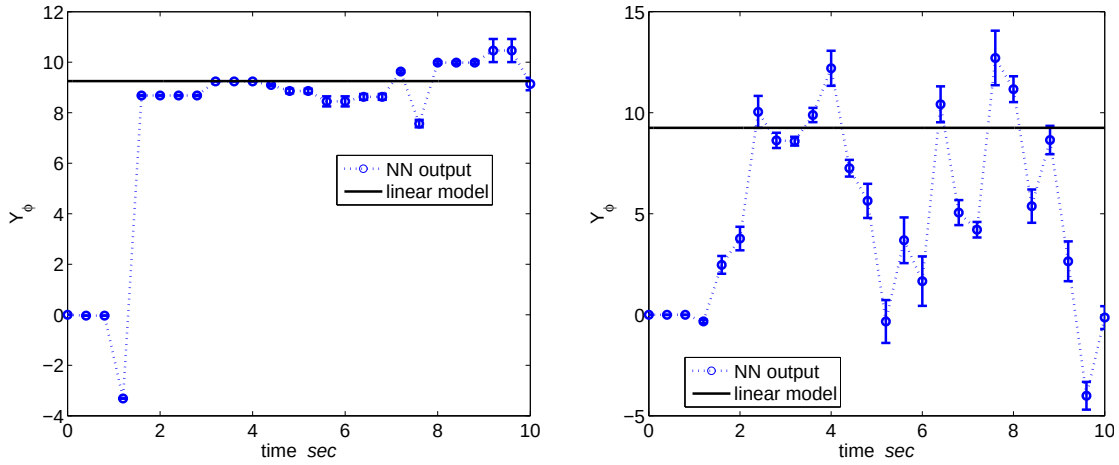

Figure 4.169: Y_ϕ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

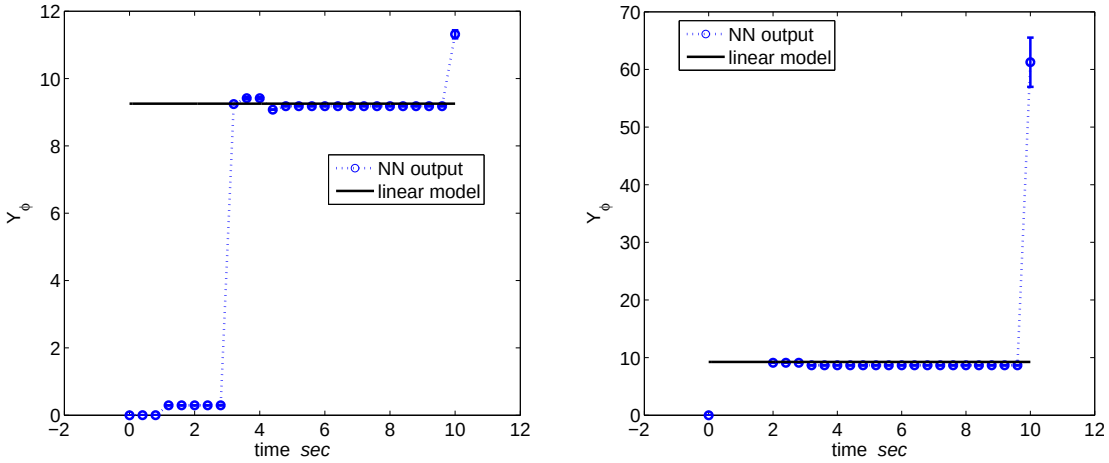

Figure 4.170: Y_ϕ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.42: Y_ϕ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	82(191)	21(91)	58(196)	92(150)
stack - 100	227(365)	232(275)	51(220)	84(211)
stack - 150	157(430)	332(351)	21(219)	64(399)

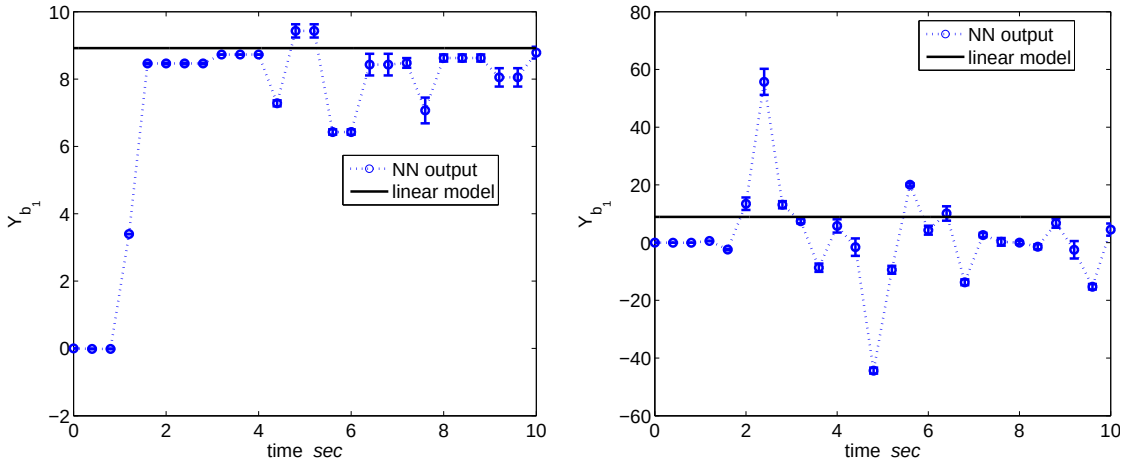

Figure 4.171: Y_{b_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

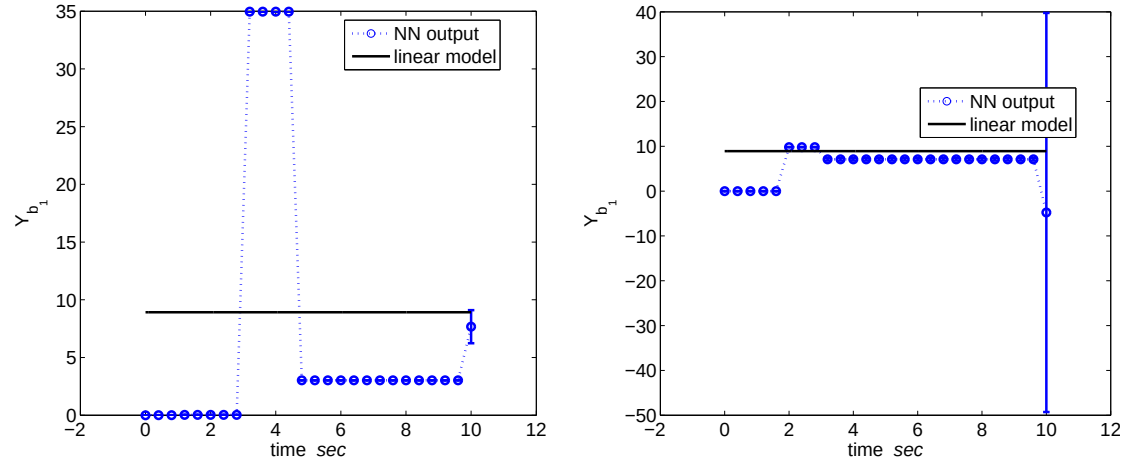

Figure 4.172: Y_{b_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.43: Y_{b_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	74(148)	2(2)	2(38)	0(17)
stack - 100	144(412)	22(57)	10(84)	29(32)
stack - 150	149(430)	0(19)	12(68)	0(399)

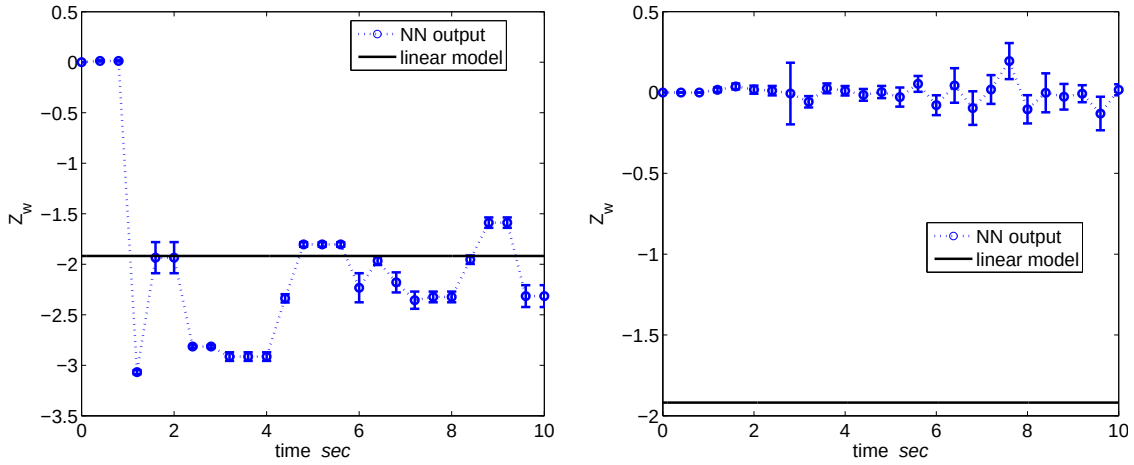

Figure 4.173: Z_w forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

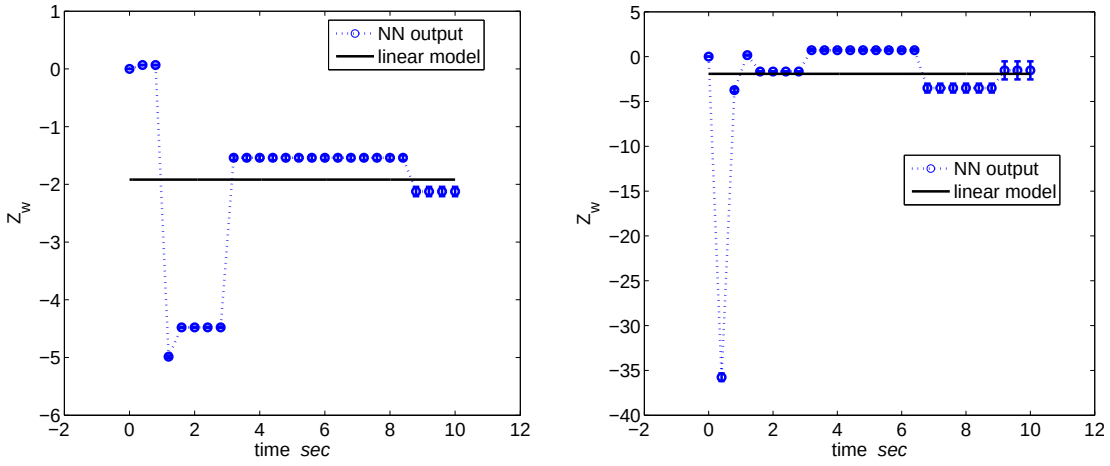

Figure 4.174: Z_w forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.44: Z_w forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	37(259)	56(151)	0(1)	1(3)
stack - 100	142(343)	0(63)	0(0)	1(5)
stack - 150	95(333)	0(351)	0(0)	0(145)

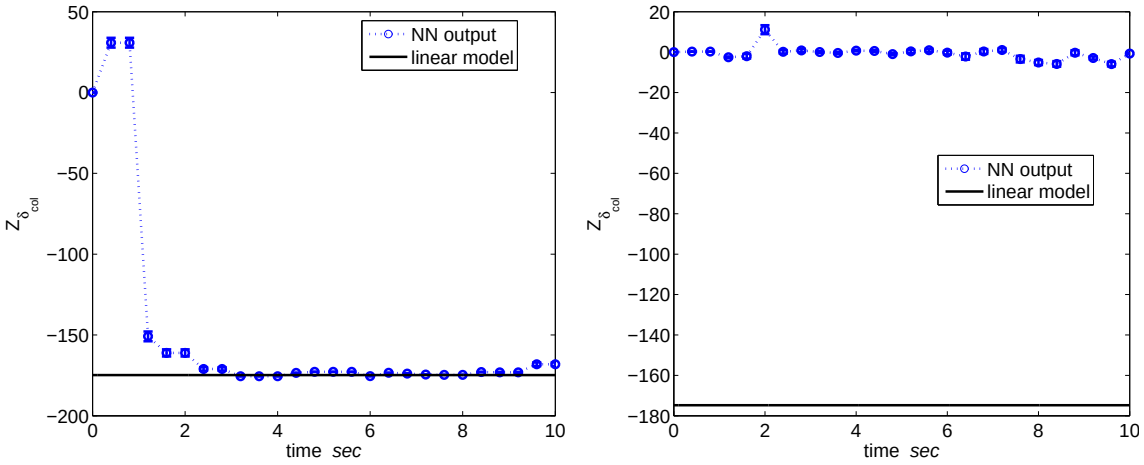

Figure 4.175: $Z_{\delta_{col}}$ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

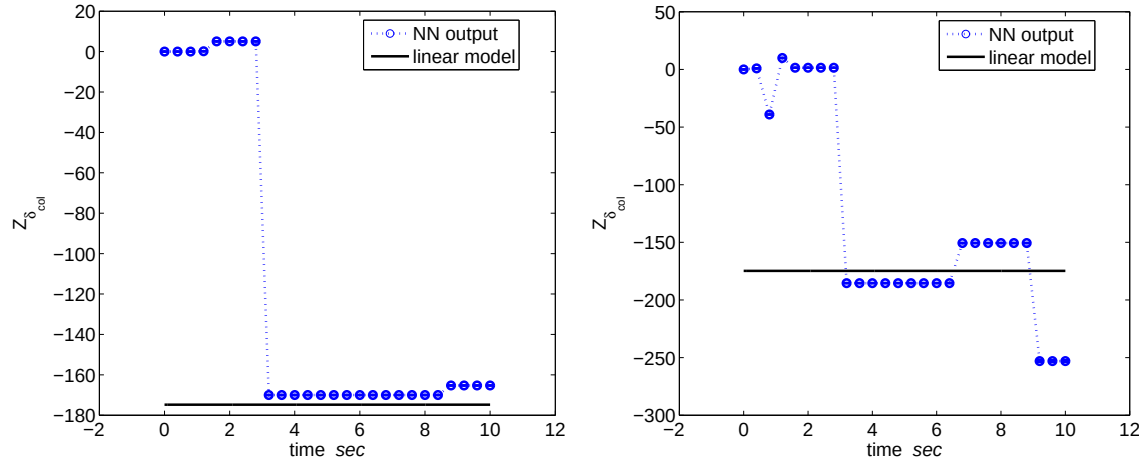

Figure 4.176: $Z_{\delta_{col}}$ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.45: $Z_{\delta_{col}}$ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	306(306)	0(0)	0(0)	1(97)
stack - 100	361(404)	60(349)	0(0)	62(405)
stack - 150	385(443)	286(351)	0(0)	0(296)

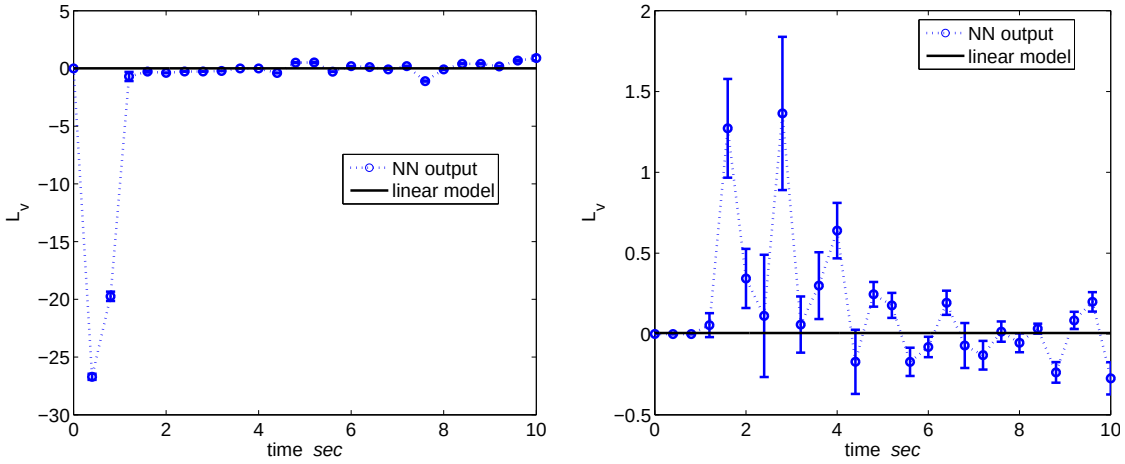

Figure 4.177: L_v forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

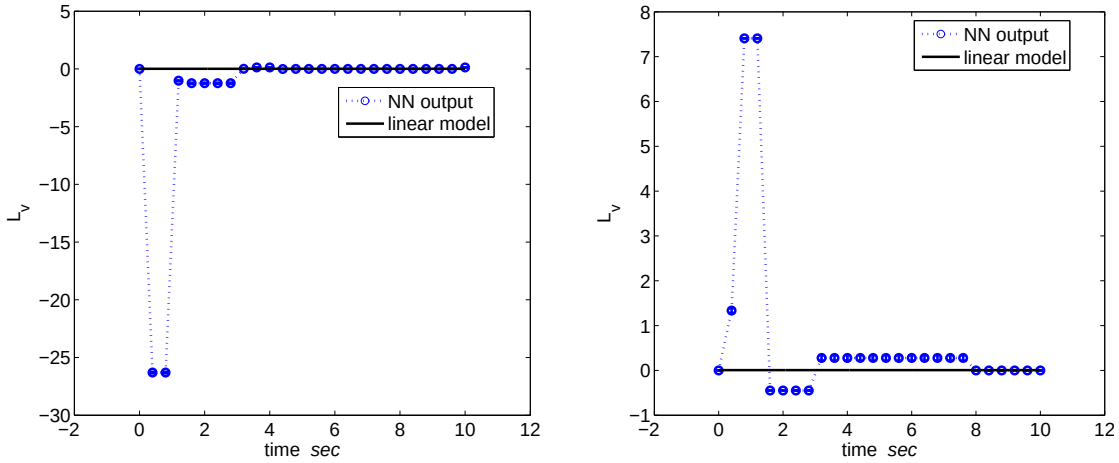

Figure 4.178: L_v forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.46: L_v forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(0)	0(0)	0(41)	0(31)
stack - 100	0(21)	0(0)	0(1)	0(0)
stack - 150	0(36)	0(0)	2(4)	0(0)

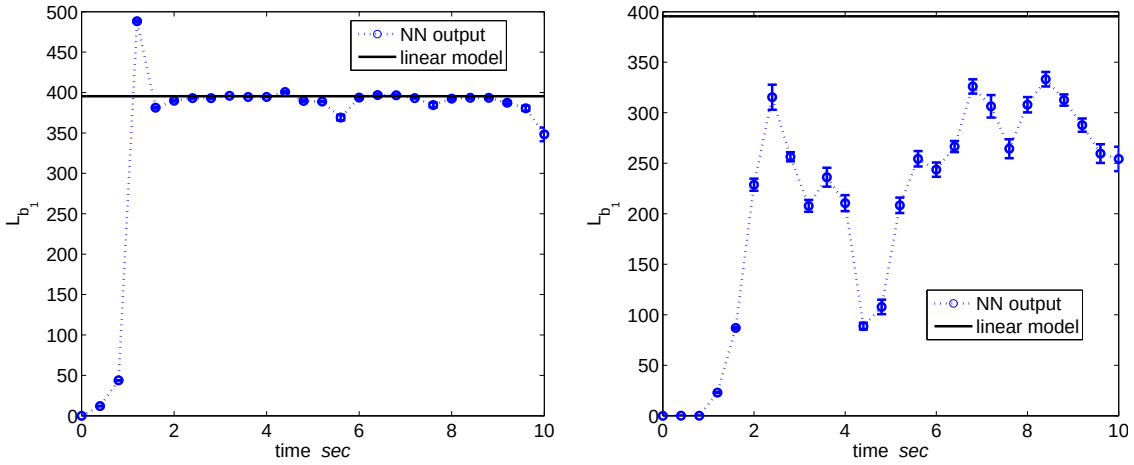

Figure 4.179: L_{b_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

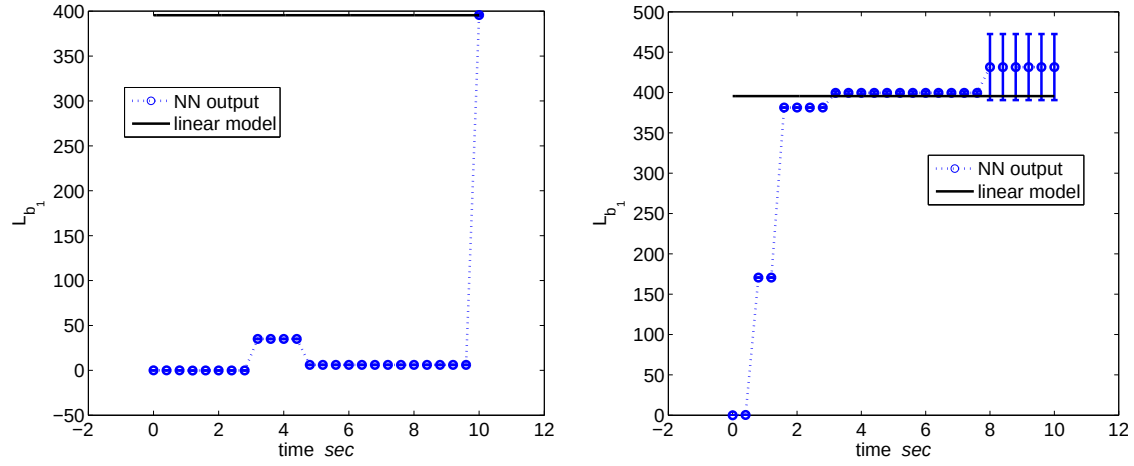

Figure 4.180: L_{b_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.47: L_{b_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	265(289)	6(6)	0(1)	69(263)
stack - 100	386(449)	91(91)	1(154)	396(458)
stack - 150	410(448)	19(19)	0(269)	333(439)

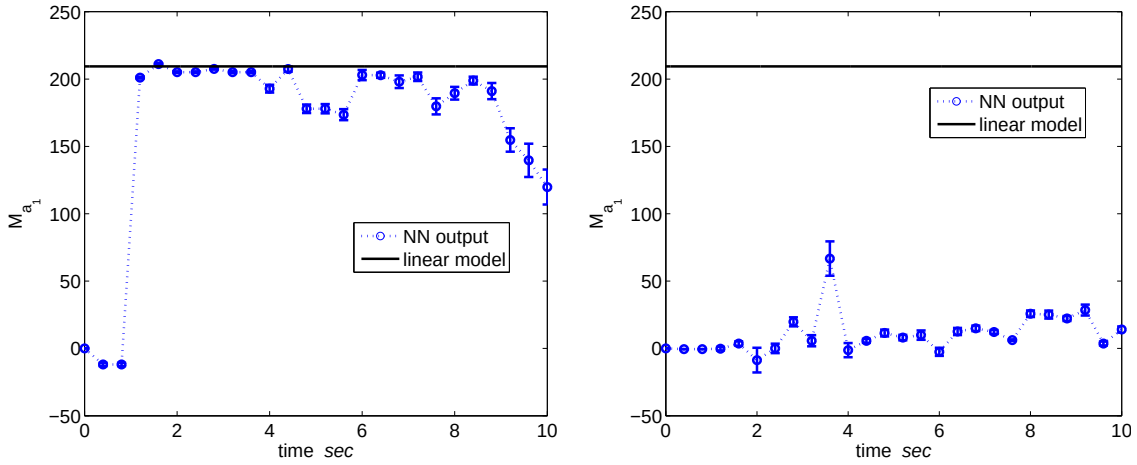

Figure 4.181: M_{a_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

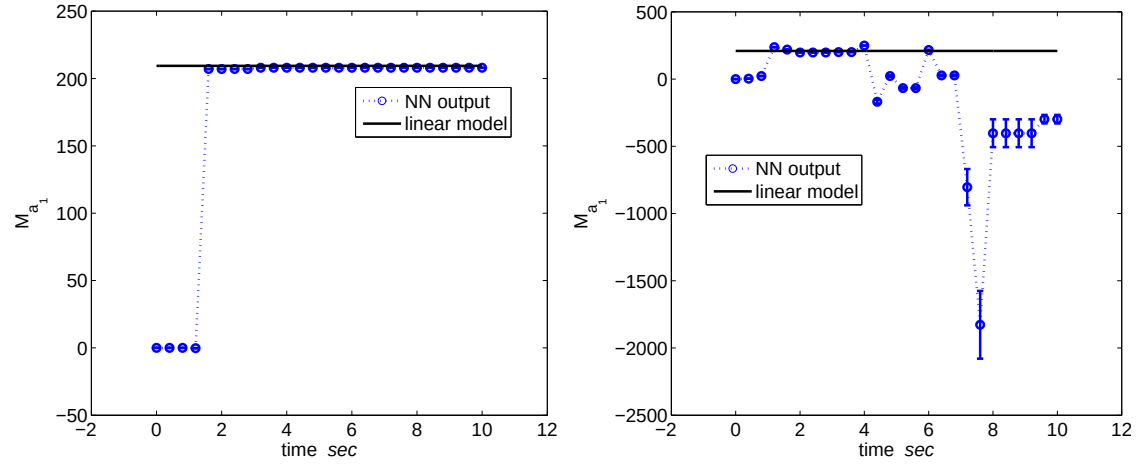

Figure 4.182: M_{a_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.48: M_{a_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	88(270)	74(130)	0(0)	8(50)
stack - 100	198(417)	105(143)	0(0)	70(135)
stack - 150	231(439)	430(430)	0(0)	72(167)

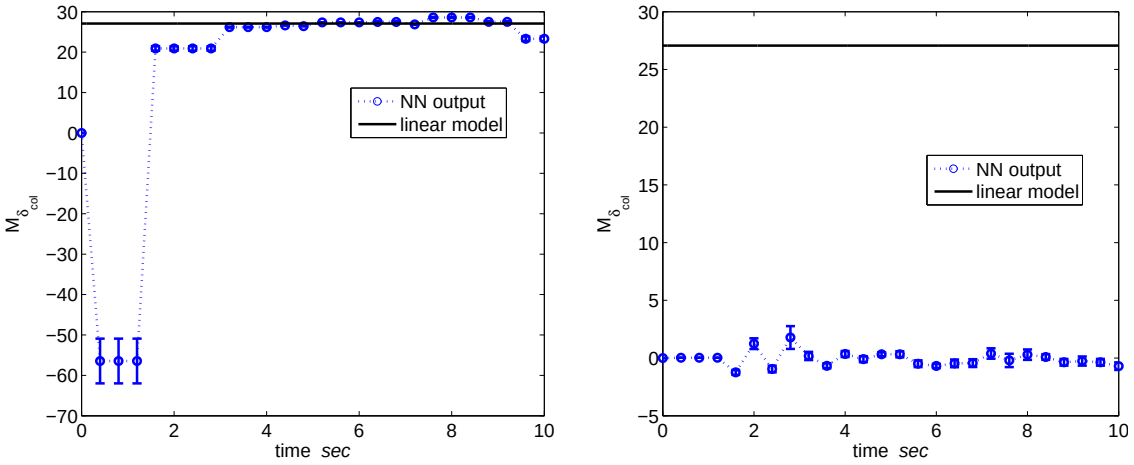

Figure 4.183: $M_{\delta_{col}}$ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

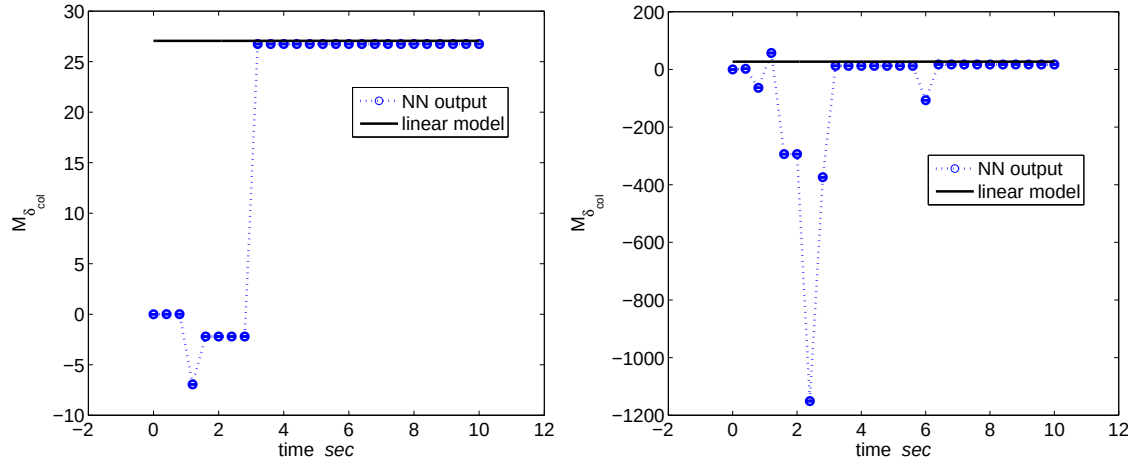

Figure 4.184: $M_{\delta_{col}}$ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.49: $M_{\delta_{col}}$ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	149(282)	7(26)	0(2)	0(57)
stack - 100	146(395)	0(91)	0(0)	2(16)
stack - 150	248(434)	351(351)	0(0)	2(187)

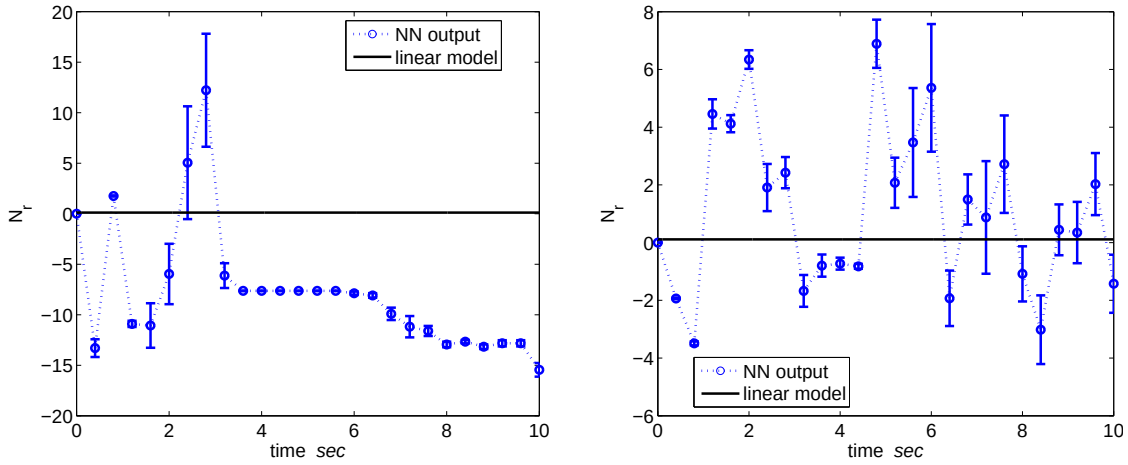

Figure 4.185: N_r forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

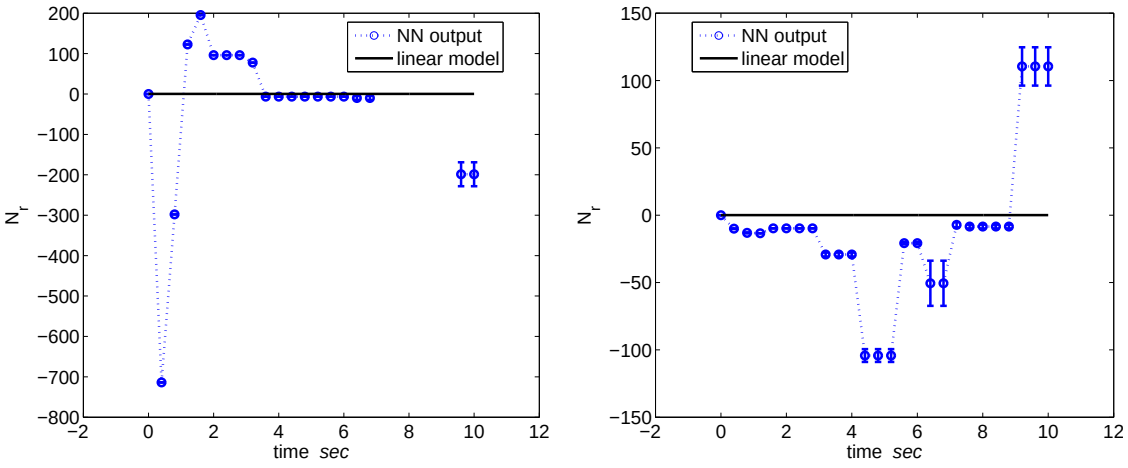

Figure 4.186: N_r forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.50: N_r forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(1)	0(0)	0(3)	0(0)
stack - 100	0(0)	0(0)	0(3)	0(0)
stack - 150	0(0)	0(0)	0(5)	0(0)

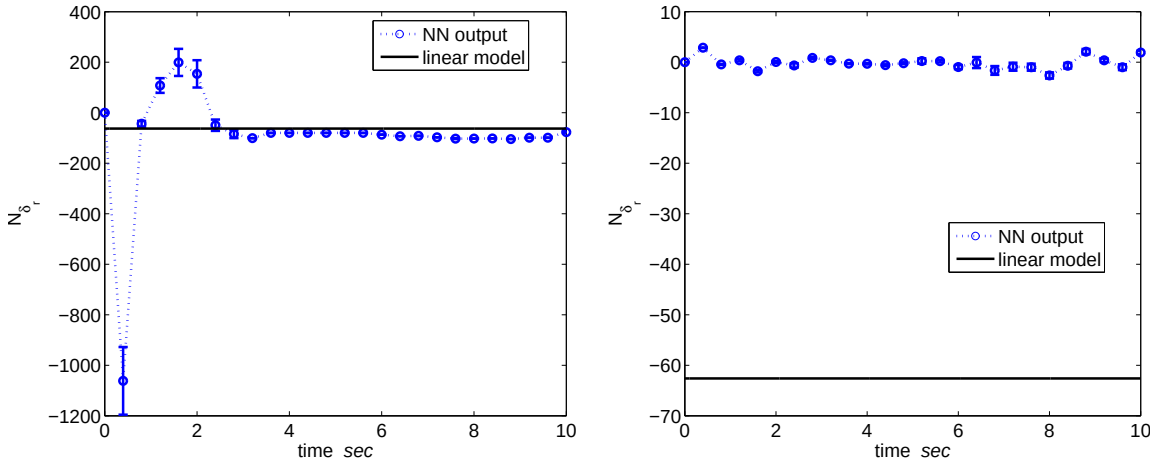

Figure 4.187: N_{δ_r} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

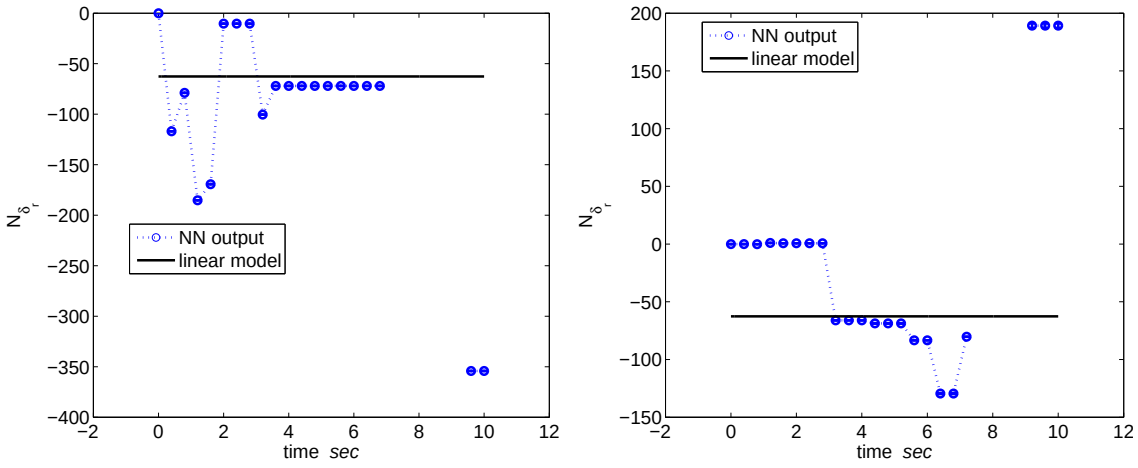

Figure 4.188: N_{δ_r} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.51: N_{δ_r} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	0(6)	0(10)	0(0)	0(5)
stack - 100	0(0)	0(0)	0(0)	0(3)
stack - 150	0(190)	0(204)	0(0)	0(152)

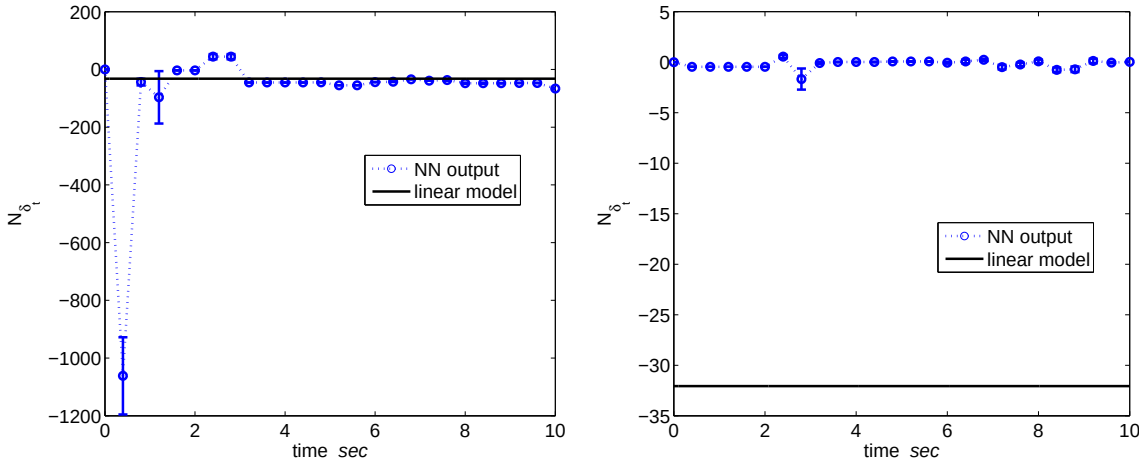

Figure 4.189: N_{δ_t} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

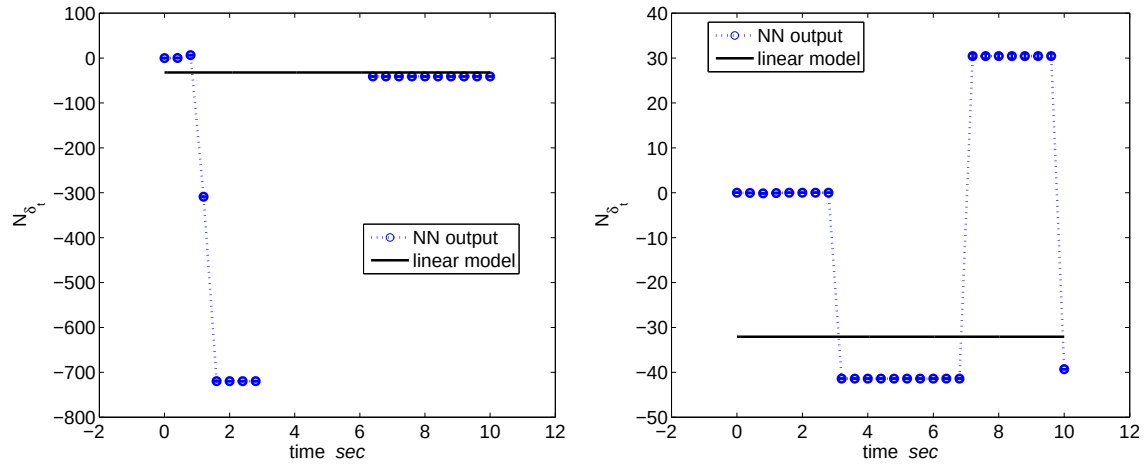

Figure 4.190: N_{δ_t} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.52: N_{δ_t} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	15(67)	0(13)	0(0)	1(9)
stack - 100	0(85)	5(75)	0(0)	0(2)
stack - 150	0(138)	0(197)	0(0)	0(219)

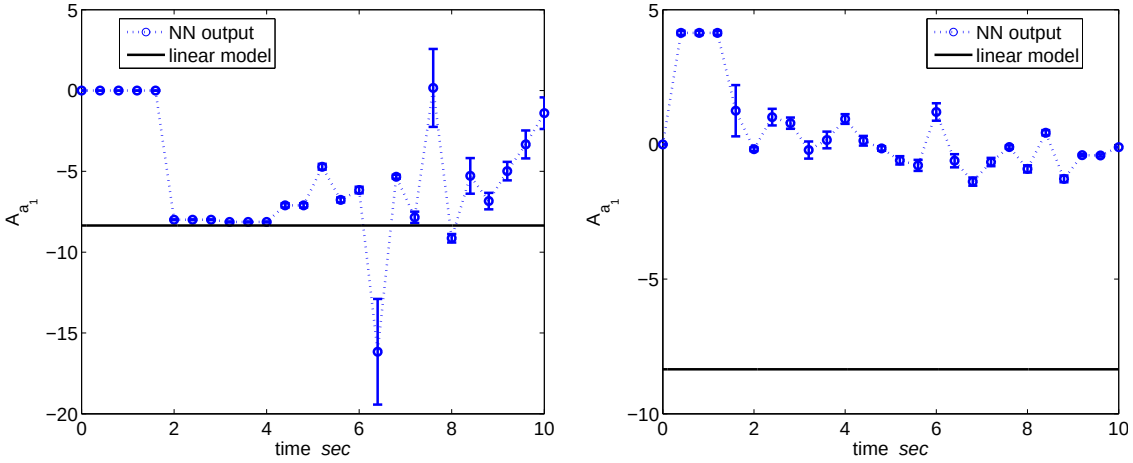

Figure 4.191: A_{a_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

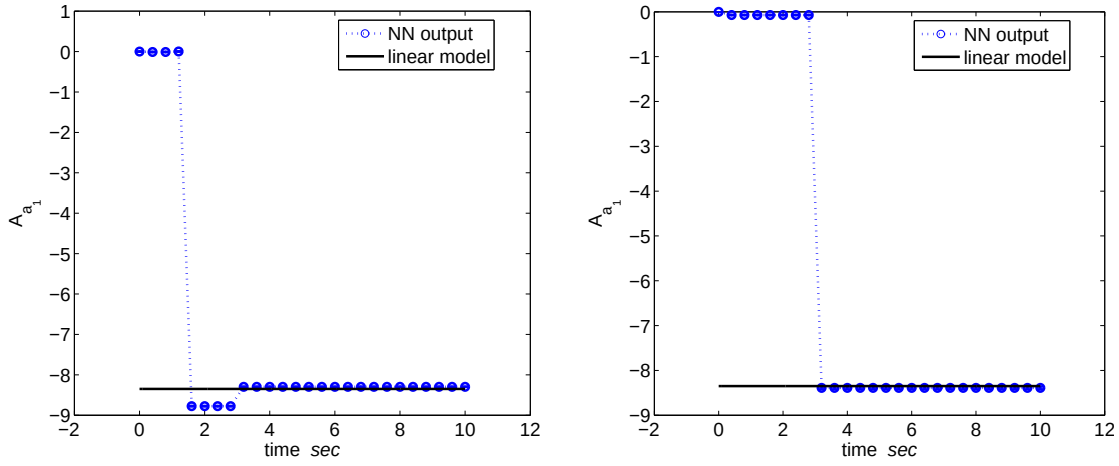

Figure 4.192: A_{a_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.53: A_{a_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	92(188)	37(66)	0(1)	0(92)
stack - 100	104(335)	105(244)	0(4)	60(60)
stack - 150	130(309)	351(430)	0(1)	351(351)

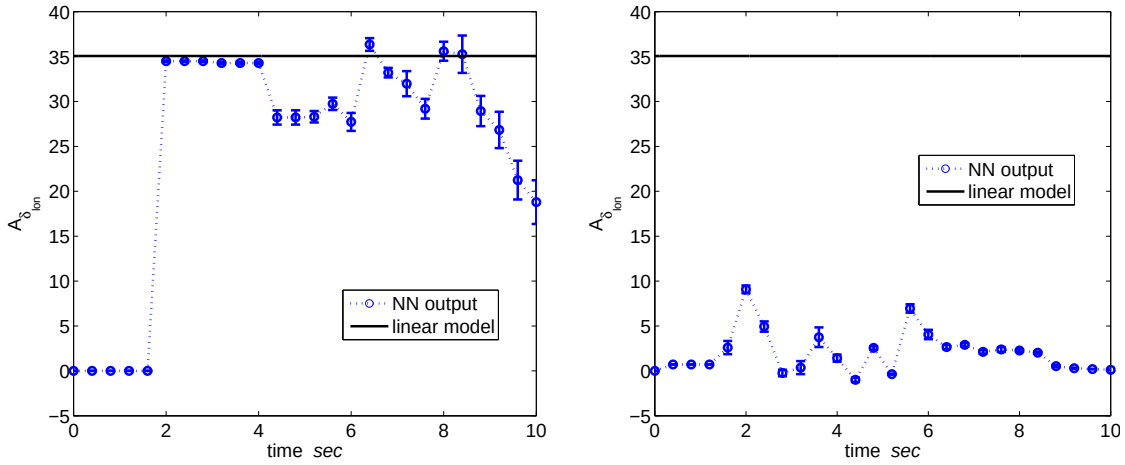

Figure 4.193: $A_{\delta_{lon}}$ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

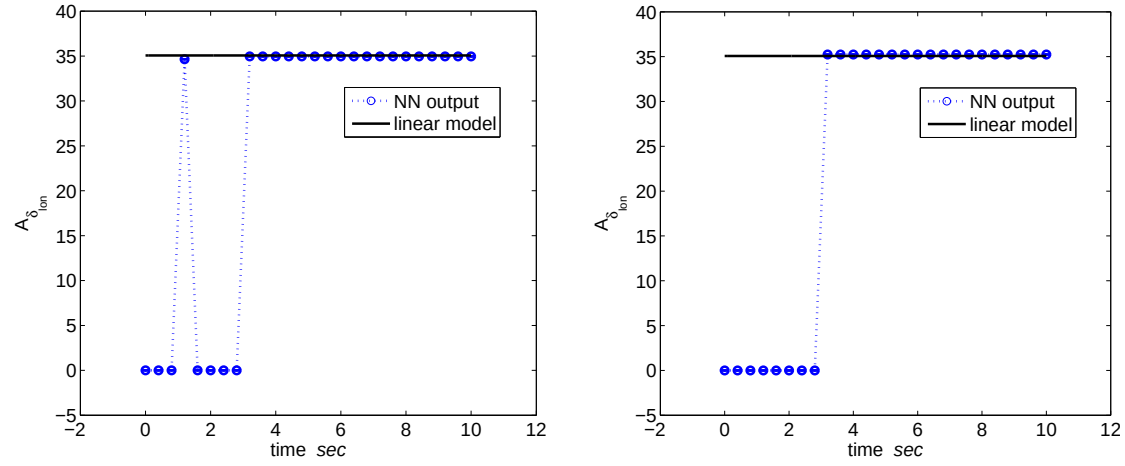

Figure 4.194: $A_{\delta_{lon}}$ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.54: $A_{\delta_{lon}}$ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	69(283)	0(0)	0(0)	0(0)
stack - 100	150(416)	0(0)	0(1)	0(0)
stack - 150	187(398)	367(367)	0(3)	351(351)

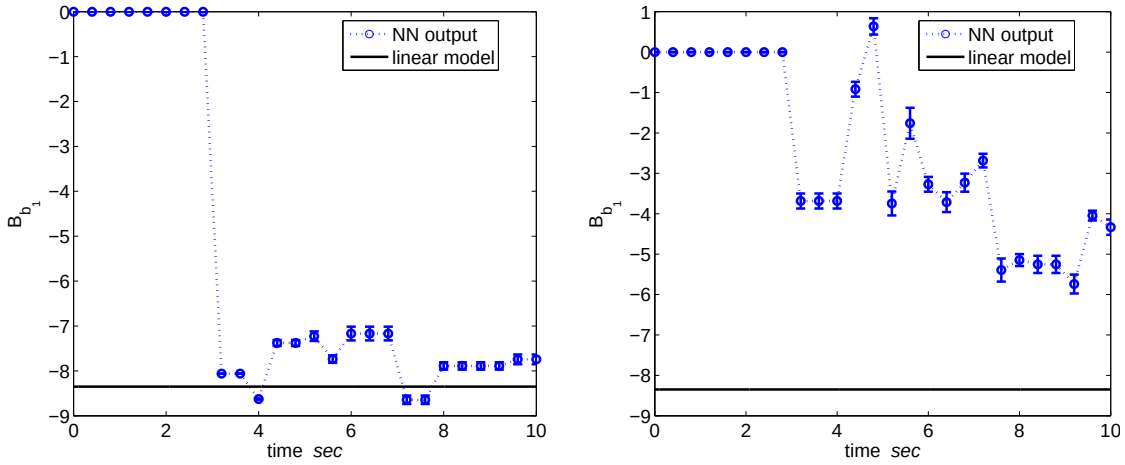

Figure 4.195: B_{b_1} forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

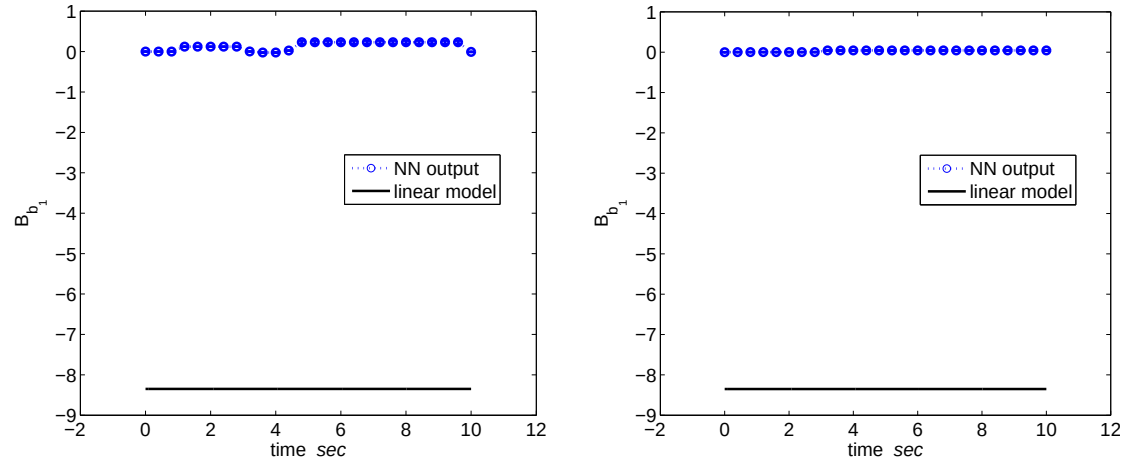

Figure 4.196: B_{b_1} forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.55: B_{b_1} forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	162(276)	0(0)	1(3)	0(0)
stack - 100	177(397)	0(0)	0(19)	0(0)
stack - 150	95(351)	0(0)	0(83)	0(0)

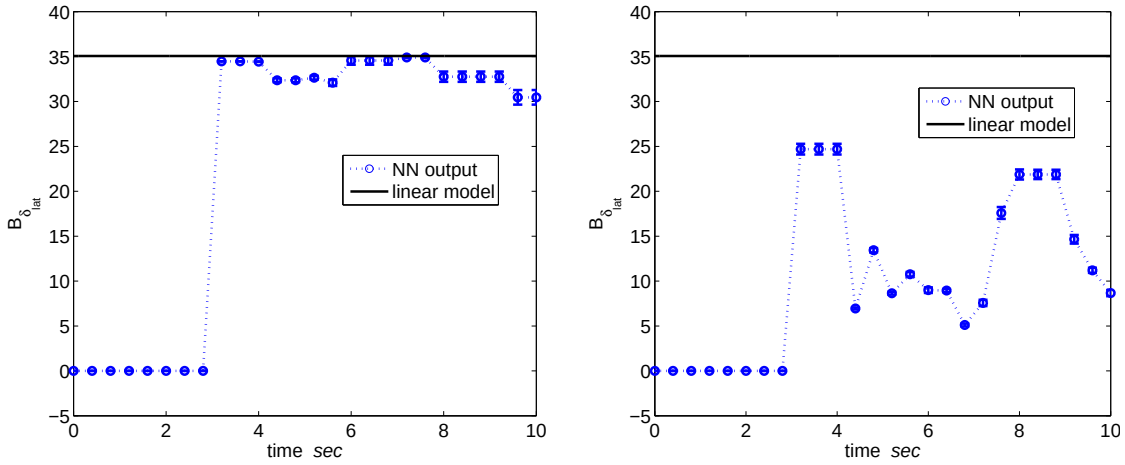

Figure 4.197: $B_{\delta_{lat}}$ forward 20m/s flight DM online estimation: (1) ideal, (2) turbulence.

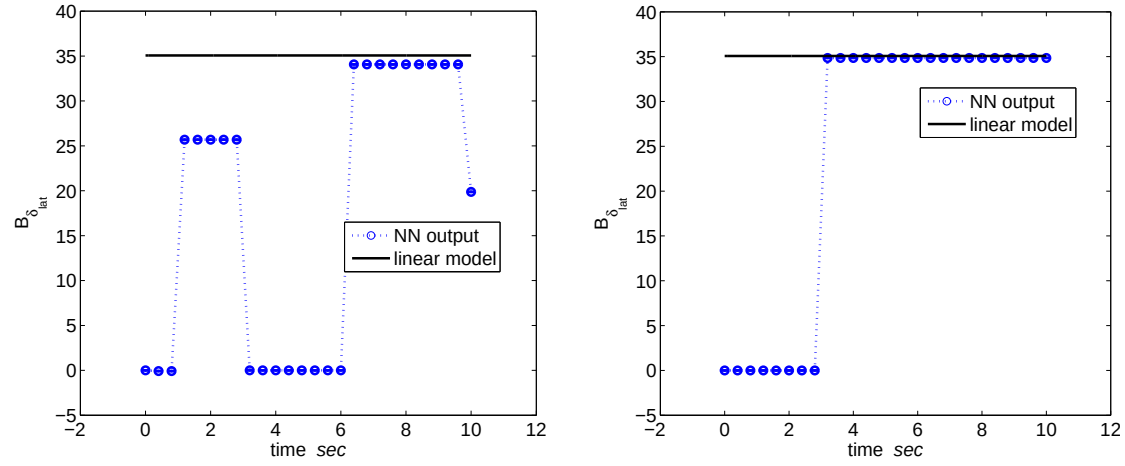

Figure 4.198: $B_{\delta_{lat}}$ forward 20m/s flight MDM online estimation: (1) ideal, (2) turbulence.

Table 4.56: $B_{\delta_{lat}}$ forward 20m/s flight: No. of estimated values with 95(60)% confidence

	ideal		**turbulence**	
	DM	MDM	DM	MDM
stack - 50	179(240)	0(28)	0(0)	0(3)
stack - 100	248(359)	60(267)	0(39)	0(36)
stack - 150	155(351)	191(279)	0(99)	351(351)

Chapter 5

Reconfigurable Flight Control Design

5.1 Reconfigurable Control

The miniature helicopter is a versatile vehicle capable of aggressive maneuvers and complex movements. Unlike their fixed-wing counterparts, helicopters have a distinct advantage of hovering and flying backwards. Autonomy of these helicopters has shown successful applications in urban reconnaissance and search and rescue missions (Suresh et al., 1995; Kumar et al., 2003). However, their performance has been limited by the design of the flight control system (Johnson and Kannan, 2005). This is due to the flight control system been designed using linearized mathematical models at various flight conditions (Savran et al., 2006). Gain scheduling has been used to provide good performance within the flight envelope although poor tracking performance has been shown under severe uncertainties due to unmodelled dynamics and general fault failures (Suresh and Kannan, 2008).

In the presence of parameter uncertainties and unmodelled dynamics, reconfiguration of the helicopter flight control law becomes desirable during different phases of flight such as: hover, high speed cruise and autorotation. In the past years, the design of aircraft flight control systems to accomplish reconfiguration, focused mainly on battle damage with loss or failure of actuator surface effectiveness (de Weerdt, 2005; Chandler et al., 1995). This approach makes use of Failure Detection Identification (FDI) and Adaptive Reconfigurable Control (ARC) models to identify which failure has occurred. A switching mechanism chooses which control actions should be used by identifying which model accurately predicted the aircraft dynamics (de Weerdt, 2005). This approach requires a controller to be designed for each failure model. To model the failures accurately, a priori knowledge of aircraft behaviour for each failure is required, which is hard to realize (de Weerdt, 2005).

5.1.1 Neural network adaptive control

In the case of a miniature unmanned helicopter, the flight controller is required to adapt to changes in the system dynamics and reconfigure its control law to guarantee stability throughout the entire flight envelope (Johnson and Kannan, 2005; Leitner et al., 1995; Munzinger, 1998; Mettler, Kanade, Tischler and Messner, 2000). Neural networks have been used to improve the performance of the flight control systems by adapting to varying dynamics and parameter uncertainties (Kar and Behera, 2009; Suresh and Kannan, 2008; Johnson and Kannan, 2005; Pesonen et al., 2004; Chuntao and Yonghong, 2006; Calise and Rysdyk, 1997; Savran et al., 2006). Johnson and Kannan (2002) developed a neural network-based direct adaptive control system which enabled adaptation to the outerloop dynamics that controls the trajectory and using Pseudo Control Hedging (PCH) to prevent adaptation of the innerloop that controls attitude. This enabled the bandwidth in the outerloop thus improving tracking performance. de Weerdt (2005) used a combination of the dynamic inversion and neural networks to develop an adaptive controller flight control reconfiguration. The neural networks were used to compensate for modelling errors introduced by the inverse controller and also provide adaptation to any kind of unanticipated failures during flight.

Direct adaptive control

The objective of a direct adaptive neural controller, shown in Figure 5.1, is to approximate the control law using neural networks such that the system response follows the reference command. The neural network input vector V is comprised of system outputs y and past control variable deflections u and the reference command y_d. The output u is the present control variable deflections represented as:

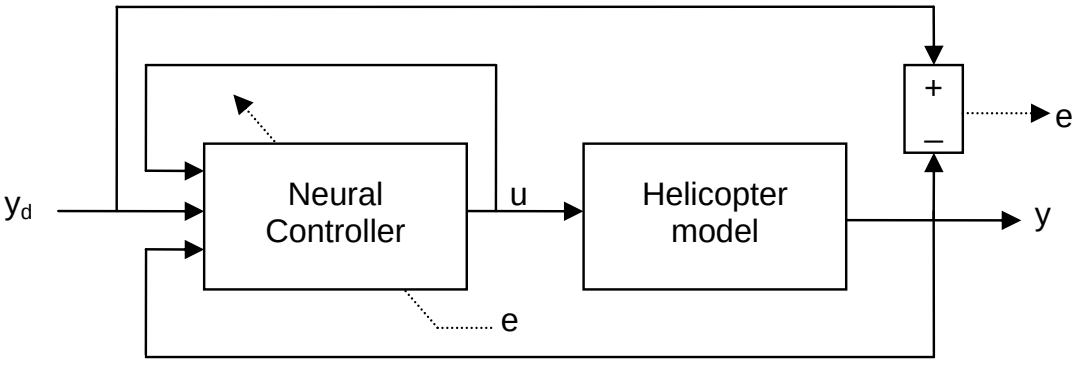

Figure 5.1: Neural Network Direct Adaptive Control System.

$$u(k) = w^y \sigma(w^h V) \tag{5.1}$$

where w^y is the weight matrix between the hidden layer and the output layer with h hidden neurons, w^h is the weight matrix between the input layer and the hidden layer and σ is the activation function in the hidden layer (Billings et al., 1992). The learning goal is to find the optimal weights such that:

$$J \overrightarrow{min} \sum_{k=1}^{N} e(k)^2 \tag{5.2}$$

where J is the cost function and $e(k)$ is the difference between the desired target and the aircraft response defined as:

$$e(k) = y_d(k) - y(k) \tag{5.3}$$

In the case of an unstable aircraft, such as the helicopter, the response of the aircraft may grow unbounded for a bounded control input (Suresh and Kannan, 2008). This will cause the state and output variables to escape to infinity. This problem is addressed by training the network off-line within a finite sequence and controller weights are adapted online for any aerodynamic uncertainty and control surface failures.

Dynamic inversion adaptive control

The method of dynamic inversion control is a form of feedback linearization which states that if an exact form of the system dynamic equations is known, and all the states are measurable, a controller can be formulated to make the input-output behaviour of the system that of a set of integrators. Given the system dynamics in the form (de Weerdt, 2005):

$$\dot{x} = F(x, u, t) \tag{5.4}$$

$$y = h(x, u, t) \tag{5.5}$$

where x is the state vector, u is the control input vector and y is the output vector.

Assuming that the exact form of the system dynamic equations are known and all the states can be measured, a controller can be formulated to make the input-output behaviour of the system that of a set of integrators. Defining a state-space system:

$$\dot{x} = Ax + Bu \tag{5.6}$$

$$y = Cx \tag{5.7}$$

$$\dot{y} = C\dot{x} = CAx + CB(u_0 + \Delta u) \tag{5.8}$$

where u_0 is the trim control input and Δu is the variable control input. Isolation of the control inputs can be performed:

$$CB\Delta u = \dot{y} - (CAx + CBu_0) \tag{5.9}$$

The objective of the dynamic inversion control law is to solve for an inversion gain matrix P such that:

$$\Delta u = P(\dot{y}_d - \dot{y}_0) \tag{5.10}$$

where

$$\dot{y}_0 = CAx + CBu_0 \tag{5.11}$$

$$\dot{y}_d = \dot{y} \tag{5.12}$$

y_0 is the initial response and y_d is the desired response from the system dynamics. Furthermore, if the plant is comprised of estimated parameters which include modelling errors, then

$$\hat{A} \neq A \tag{5.13}$$

$$\hat{B} \neq B \tag{5.14}$$

to compensate for modelling errors, Equation 5.12 can be augmented as

$$\dot{y} = \hat{f}(x, \Theta) + \dot{y}_d \tag{5.15}$$

where Θ depend on the modelling errors such that

$$\Theta = [\theta_1 \dots \theta_n]^T \tag{5.16}$$

where n is the size of the state vector. Equation 5.15 forms part of a linear regression model and can be solved as described in Section 4.5.2. Thus, RBF-based neural networks are then integrated as the adaptive element to solve for the modelling errors.

5.2 Controller Development

The proposed structure of a NN-based control system is shown in Figure 5.2. The system dynamics are identified using neural networks as described in Section 4.4. The parameter estimation is then performed to extract aerodynamic derivatives as described in Section 4.5. These parameters are used in the dynamic inversion control law and the neural networks are used to compensate for modelling errors introduced by these estimated parameters. This

forms the basis of a neural network reconfigurable flight controller (NN-RFC), which retains stability as the aircraft moves away from its operating point. This is achieved as neural networks are used to compensate for the increasing modelling errors developed by the dynamic inversion law.

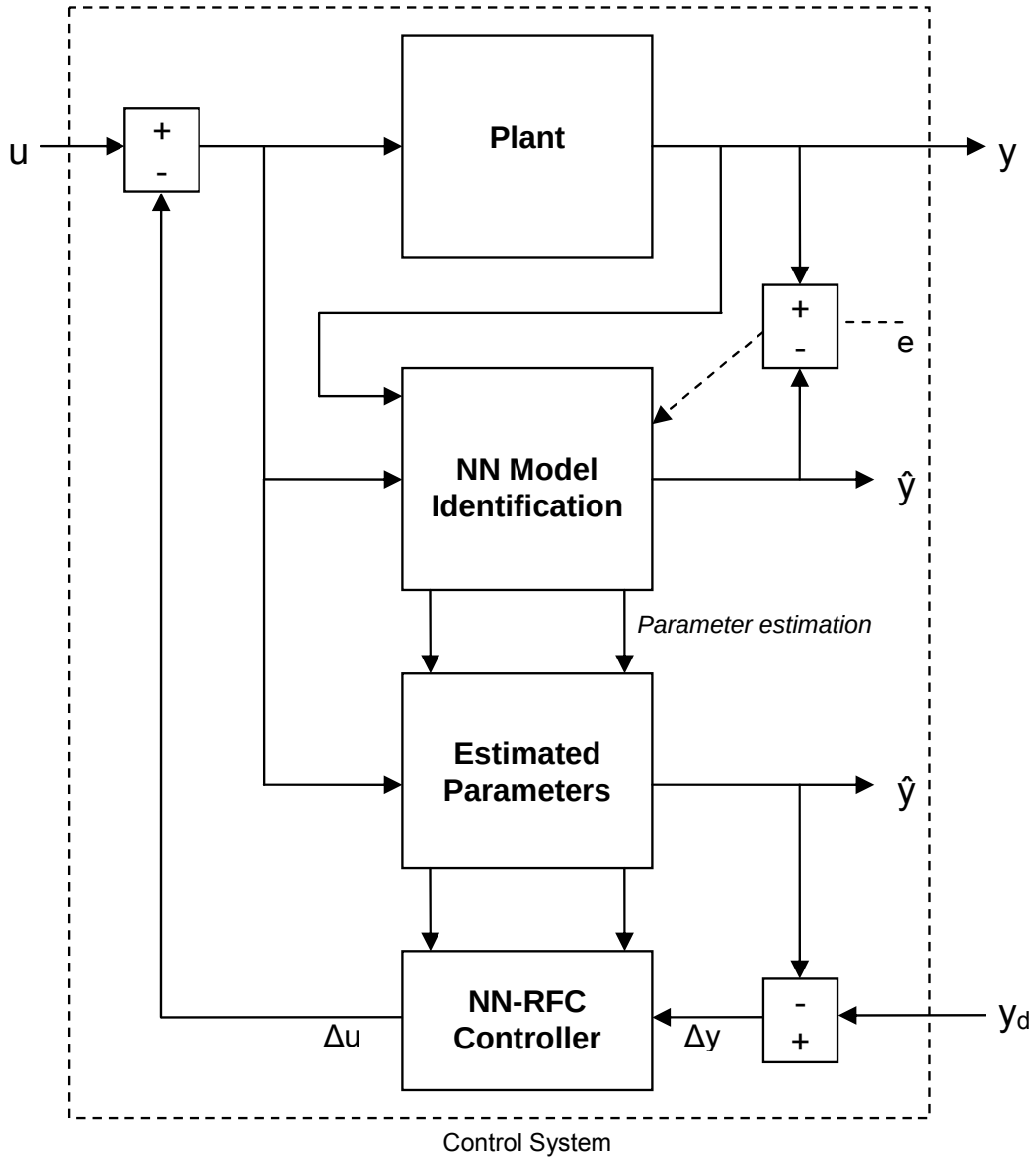

Figure 5.2: Structure of NN-RFC Control System

A miniature helicopter state-space system can be described

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{a_1} \\ \dot{b_1} \end{bmatrix} = \hat{A} \begin{bmatrix} u \\ v \\ w \\ \phi \\ \theta \\ p \\ q \\ r \\ a_1 \\ b_1 \end{bmatrix} + \hat{B} \begin{bmatrix} \delta_{lon} \\ \delta_{lat} \\ \delta_{col} \\ \delta_r \\ \delta_t \end{bmatrix} \tag{5.17}$$

where $\hat{A}$ and $\hat{B}$ are the estimated system and control matrices respectively. These matrices are comprised of the estimated parameters from Chapter 4. The input vector deflections can thus be computed based on the dynamic inverse adaptive control law:

$$\begin{bmatrix} \Delta\delta_{lon} \\ \Delta\delta_{lat} \\ \Delta\delta_{col} \\ \Delta\delta_r \\ \Delta\delta_t \end{bmatrix} = (C\hat{B})^{-1} \left(\hat{f}(x,\Theta) + \dot{y}_d - \left(C\hat{A} \begin{bmatrix} u \\ v \\ w \\ \phi \\ \theta \\ p \\ q \\ r \\ a_1 \\ b_1 \end{bmatrix} + C\hat{B} \begin{bmatrix} \delta_{lon_0} \\ \delta_{lat_0} \\ \delta_{col_0} \\ \delta_{r_0} \\ \delta_{t_0} \end{bmatrix} \right) \right) \tag{5.18}$$

5.2.1 Attitude control

Attitude control for the R-50 small-scaled helicopter has been studied (Mettler et al., 2000). The identified model of the vehicle dynamics explicitly accounts for the coupled rotor/ stabiliser/fuselage dynamics. The same approach will be used here. It is assumed that correct model identification and parameter estimation was performed to obtain the aerodynamic parameters for matrices $\hat{A}$ and $\hat{B}$. These matrices are given in Section 4.5.6. The MATLAB/SIMULINK control system setup is given in Figure 5.3. For attitude control design, it has been shown that the attitude loop needs to use the attitude angles and the attitude rates for feedback (Mettler et al., 2000). The system desired output can therefore be defined:

$$y_d = [\phi \;\; \dot{\phi} \;\; \theta \;\; \dot{\theta}] \tag{5.19}$$

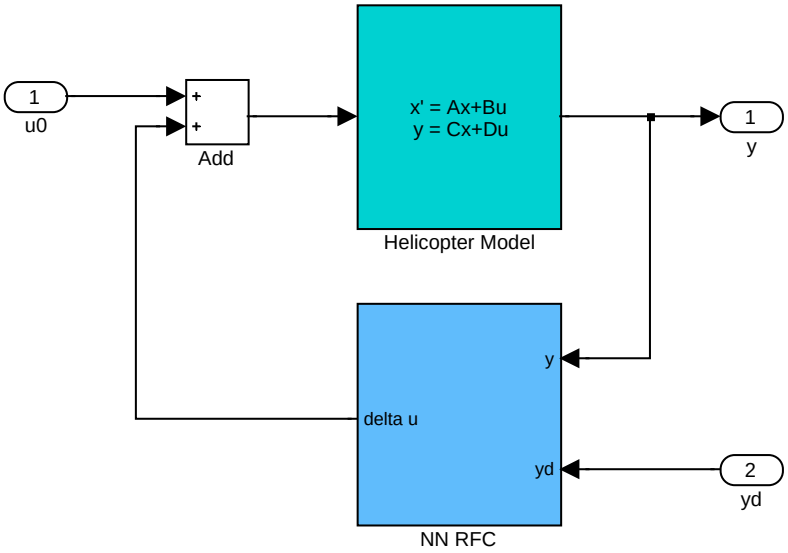

Figure 5.3: NN-RFC Control System Setup

where ϕ and θ are the roll and pitch angles respectively. The control inputs vector comprises of the cyclic inputs $\delta_{lon}, \delta_{lat}$, collective δ_{col}, yaw pedal δ_r and throttle level δ_t defined as:

$$u = [\delta_{lon} \;\; \delta_{lat} \;\; \delta_{col} \;\; \delta_r \;\; \delta_t] \tag{5.20}$$

5.2.2 Attitude control system requirements

The change of the helicopter attitude has a direct effect on the longitudinal and lateral maneuvers of the helicopter, thereby making attitude control performance essential for the overall helicopter performance. The following control requirements are specified (Mettler et al., 2000):

- **Speed of response**: High bandwidth is required for good handling qualities. For full-scale helicopters, the specification for a level 1 handling quality is around 2 rad/sec. This also affects tracking (velocity and position) control.
- **Sufficient damping of the rotor/stabiliser bar/fuselage mode**: to limit the short period roll and pitch oscillations.
- **Disturbance rejection**: atmospheric turbulence act as input disturbances which can be rejected through high bandwidth attitude control.

5.3 Controller Validation

The validation process verifies how well the developed control system 'tracts' reference commands for pitch θ_{ref} and roll ϕ_{ref} and the effects of the cross-coupling effects in the various flight phases. To further determine conroller performance, gain and phasis margin analysis is required. It is expected that the lightly damped rotor/stabilizer/fuselage modes, shown in Section 3.3, constitutes a performance and robustness limitation. To optimize the attitude control system, the following specifications have being used (Mettler et al., 2000):

- **Eigenvalue location:** All real parts of the system eigenvalues must less or equal to zero in all flight phases.
- **Gain and phase margins:** a minimum of 45 deg of phase margin and 6dB of gain margin must be satisfied (MIL-F-9490).
- **Crossover frequency:** to reduce control activity, the crossover frequency limit is set at a Level 1 flying qualities of 10 rad/s.

It is expected to see different performance levels for each flight phase and input disturbance. The RBF network memory usage is expected to increase as a consequence to a trade-off between controller performance and system robustness.

Chapter 6

Conclusion and Recommendations

The development of the miniature helicopter simulation model was achieved using the MATLAB/ SIMULINK environment. The equations of the motion was a combination of classical rigid body and rotor dynamics which included the effect of the stabilizer bar. This was done by lumping the main rotor and stabilizer bar flapping dynamics into tip-path plane flapping dynamics. The aerodynamic model was subjected to a Dryden Spectrum model to explore the effects of atmospheric turbulence on the measured variables. The effect of slow cruise speed ($V \approx 10m/s$) on the helicopter dynamics was achieved through modelling main/tail rotor wake interactions.

The trim and stability analysis was achieved for three trim conditions: hover flight, 10m/s forward flight and 20m/s forward flight. Strong main/tail wake interactions resulted in the forward 10m/s forward flight conditions as the natural modes were strongly damped. The large perturbations in helicopter dynamics occurred due to the Phugoid mode, which transitioned from an unstable to stable mode as the forward flight speed increases. The relatively constant coupled rotor/stabilizer/fuselage modes is an indication of the rigid rotor that characterizes miniature helicopters.

The use of the RBFN over MLPN network for parameter estimation was proven as the use of the spread constant enabled the RBF-based network to accurately predict the underlying dynamics of the system. The NN-based model identification produced satisfactory results although the presence of atmospheric turbulence and sensor noise had an adverse effect on network size and memory usage. The RBF networks showed good robustness to noise for all flight conditions but this is heavily dependent on good signal-to-noise ratios.

The Delta Method (DM) and the Modified Delta Method (MDM) was investigated for the NN-based online estimation of aerodynamic parameters. It can be clearly seen that the MDM method complimented the RBF-based identification approach with good robustness to noise. This could be evaluated by an increasing number of estimated parameters with high confidence as the helicopter transitions from hover to forward flight. This was especially the case for the force and moment derivatives in both longitudinal and lateral dynamics. The for-

ward 10m/s flight condition produced the least number of high confidence parameters. This could be due to the strong main/tail rotor wake interactions that might occur in that flight region. This is contrary to practical observations whereby forwards speeds higher than 5m/s, generaly produce good estimation results. Moreover, poor signal content and unmodelled dynamics prevented the accurate estimation of the yaw damping derivative.

The implementation of the neural-based adaptive control using an model inversion technique allows the use of the identified parameters in computation the required control signal to minimize the error between the plant output and the desired output. The neural networks act as error compensators introduced by the inversion method. The combination of the online estimation of the aerodynamic parameters and adaptive control results in a reconfigurable control law robust to aerodynamic uncertainties due to unmodelled dynamics and general fault failures.

6.1 Recommendations

- In this study, RBF networks were used for online model identification and parameter estimation. A fixed neuron spread of 0.8326 was used and the goal/threshold value was determined using trial and error. Using an optimization method for the neuron spread and the goal value could improve network performance and possibly decrease convergence time. This approach could also prevent the network to saturate when the signal-to-noise ratio is too low for accurate model identification and parameter estimation.

- An offline-online approach should be investigate for the accurate estimation of the yaw dynamics. The identification of the yaw feedback rate and the controller gains is essential for successful identification and parameter estimation.

- The investigation into a variable moving window optimized to maximize high confidence estimated values thereby increase the bandwidth of the reconfigurable control system in the presence of sensor noise and atmospheric turbulence.

- A study into an alternative to neural-based adaptive control to compensate for modelling inversion errors can further increase the reliability of miniature unmanned helicopters for urban service and their integration into civilian airspace.

References

Billings, S. A. and Chen, S. (1992), 'Neural networks for nonlinear dynamic system modelling and identification', *International Journal of Control* **56**(1), 319–346.

Billings, S. A., Jamaluddin, H. B. and Chen, S. (1992), 'Properties of neural networks with applications to modelling non-linear dynamical systems', *International Journal of Control* **55**(1), 193–224.

Calise, A. J. and Rysdyk, R. T. (1997), Nonlinear adaptive flight control using neural networks, Technical Report 91-09-01, Georgia Institute of Technology.

Chandler, P., Patcher, M. and Mears, M. (1995), 'System identification for adaptive and reconfigurable control', *AIAA Journal of Guidance, Control and Dynamics* **18**, 516–524.

Chen, S. and Billings, S. A. (1989), 'Recursive prediction error estimator for nonlinear models', *International Journal of Control* **49**(2), 569–594.

Chen, S., Cowan, C. F. N. and Grant, P. M. (1991), 'Orthogonal least squares learning algorithm for radial basis function networks', *IEEE Transactions on Neural Networks* **2**(2), 302–309.

Chuntao, L. and Yonghong, T. (2006), 'Adaptive control of system with hysteresis using neural networks', *Journal of Systems Engineering and Electronics* **17**(1), 163–167.

Day, D. (2005), *Flying Model Helicopters*, Special Interest Model Books Ltd, Poole, Dorset.

de Weerdt, E. (2005), Neural network aerodynamic model identification for aerospace reconfiguration, Master's thesis, Delft University of Technology.

Demuth, H. and Beale, M. (2002), *Neural Network Toolbox User's Guide - For Use with MATLAB*.

Gavrilets, V. (2003), Autonomous aerobatic maneuvering of miniature helicopters, Technical report, Department of Aeronautics and Astronautics, MIT, Cambridge.

Gavrilets, V., Mettler, B. and Feron, E. (2003), Dynamic model for a miniature aerobatic helicopter, Technical report, Department of Aeronautics and Astronautics, MIT, Cambridge.

Guo, Y., Jiang, B. and Xu, Y. (2010), 'Multimodel-based flight control system reconfiguration control in the presence of input constraints', *Journal of Control Theory and applications* **8**, 418–424.

Jafari, M. R., Alizadeh, T., Gholami, M., Alizadeh, A. and Salahshoor, K. (2007), On-line identification of non-linear systems using adaptive rbf-based neural network, *in* 'Proceedings of the World Congress on Engineering and Computer Science', San Francisco, USA.

Jategaonkar, R. V. (2000), 'Bounded-variable gauss-newton algorithm for aircraft parameter estimation', *Journal of Aircraft* **37**(4), 742–744.

Jategaonkar, R. V. (2006), *Flight Vehicle System Identification - A Time Domain Methodology*, Vol. 216, American Institute of Aeronautics and Aeronautics.

Jategaonkar, R. V. (2008), Aerodynamic modelling and system identification from flight data - recent applications at dlr, *in* '2nd SAIAS symposium', South Africa.

Johnson, E. N. and Kannan, S. K. (2002), Adaptive flight control for an autonomous unmanned helicopter, *in* 'AIAA Guidance, Navigation, and Control Conference and Exhibit', AIAA-2002-4439, Montery, CA.

Johnson, E. N. and Kannan, S. K. (2005), 'Adaptive trajectory control for autonomous helicopters', *AIAA Journal of Guidance, Control, and Dynamics* **28**, 524–538.

Jun, M., Roumeliotis, S. I. and Sukhatme, G. S. (1998), State estimation of an autonomous helicopter using kalman filtering, Technical report, Institute for Robotics and Intelligent Systems, University of Southern California.

Kaletka, K. and Hamel, P. G. (1997), 'Advances in rotorcraft system identification', *Progress of Aerospace Sciences* **33**, 259–284.

Kar, I. and Behera, L. (2009), 'Direct adaptive neural control for affine nonlinear systems', *Applied Soft Computing* **9**, 756–764.

Kenne, G., Ahmed-Ali, T., Lamnabhi-Lagarrigue, F. and Nkwawo, H. (2006), 'Nonlinear systems parameters estimation using radial basis function network', *Control Engineering Practice* **14**(1), 819–832.

Kim, S. K. and Tilbury, D. M. (2004), 'Mathematical modelling and experimental identification of an unmanned helicopter robot with flybar dynamics', *Journal of Robotic Systems* **21**(3), 95–116.

Kumar, M. V., Suresh, S., Omkar, S. N., Mani, V., Ganguli, R. and Sampath, P. (2003), Identification of helicopter dynamics based on flight data using recurrent neural networks - a comparative study, *in* 'American Helicopter Society 59th Annual Forum', Phoenix, Arizona.

Kumar, R., Ganguli, R., Omkar, S. N. and Kumar, M. V. (2008), Rotorcraft parameter estimation from real time flight data using radial basis function networks, Technical report, Indian Institute of Science, India.

Leitner, J., Calise, A. and Prasad, J. V. R. (1995), Analysis of adaptive neural networks for helicopter flight controls, Technical report, Georgia Institute of Technology.

Ljung, L. (1997), *System Identification Toolbox For use with MATLAB - User's guide*.

Ljung, L. and Soberstrom, T. (1983), *Theory and Practice of Recursive Identification*, MIT Press.

Lombaerts, T., Huisman, H., Chu, Q., Mulder, J. and Joosten, J. (2009), 'Nonlinear reconfiguring flight control based on online physical model identification', *AIAA Journal of Guidance, Control and Dynamics* **32**, 727–748.

Lorenz, S. and Chowdhary, G. (2005), Non-linear model identification for a miniature rotorcraft - preliminary results, *in* 'American Helicopter Society', 61th Annual Forum.

McLean, D. (1990), *Automatic flight control systems*, Prentice Hall International (UK) Ltd.

McNally, B. D. and Bach Jr., R. E. (1988), Flight testing a v/stol aircraft to identify a full-envelope aerodynamic model, Technical Report NASA-TM-100996, NSA.

Mehra, R. K., Stepner, D. E. and Tyler, J. S. (1974), 'Maximum likelihood identification of aircraft stability and control derivatives', *Journal of Aircraft* **11**(2), 81–89.

Mettler, B. (2003), *Identification Modelling and Characteristics of Miniature Rotorcraft*, Kluwer Academic Publishers, Norwell, MA.

Mettler, B., Kanade, T. and Tischler, M. B. (1999), System identification of small-size unmanned helicopter dynamics, *in* 'Journal of the American Helicopter Society, 55th Forum'.

Mettler, B., Kanade, T. and Tischler, M. B. (2000), System identification of a model-scale helicopter, Technical Report CMU-RI-TR-00-03, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Mettler, B., Kanade, T., Tischler, M. B. and Messner, W. (2000), Attitude control optimization for a small-scale unmanned helicopter, *in* 'AIAA Guidance, Navigation and Control Conference and Exhibit', Denver, CO, pp. 1–10.

Morelli, E. A. (2000), 'Real-time parameter estimation in the frequency domain', *AIAA Journal of Guidance, Control, and Dynamics* **23**(5), 812–820.

Munzinger, C. (1998), Development of a real-time flight simulator for an experimental model helicopter, Master's thesis, Georgia Institute of Technology.

Padfield, G. D. (1996), *Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modeling*, AIAA Education Series, Reston, VA.

Pallett, T. J. and Ahmad, S. (1991), Real-time helicopter flight control: Modelling and control by linearization and neural networks, Technical Report TR-EE 91-35, School of Electrical Engineering, Purdue University.

Paris, A. C. and Bonner, M. (2004), 'Nonlinear model development from flight-test data for f/a-18e super hornet', *Journal of Aircraft* **41**(4), 692–702.

Park, J. and Sandberg, I. W. (1991), 'Universal approximation using radial basis function networks', *Neural Computation* **3**, 246–257.

Pesonen, U. J., Steck, J. E., Rokhsaz, K., Bruner, S. and Duerksen, N. (2004), 'Adaptive neural network inverse controller for general aviation safety', *AIAA Journal of Guidance, Control, and Dynamics* **27**(3), 434–443.

Polycarpou, M. and Ioannou, P. A. (1991), Identification and control of nonlinear systems using neural networks models: Design and stability analysis, Technical Report 91-09-01, Department of Electrical Engineering, University of Southern California.

Prouty, R. W. (1986), *Helicopter Performance, Stability and Control*, PWS Publishers.

Raisinghani, S. C. and Ghosh, A. K. (2001), 'Frequency-domain estimation of parameters from flight data using neural networks', *AIAA Journal of Guidance, Control and Dynamics* **24**(3), 525–530.

Raisinghani, S. C., Ghosh, A. K., and Kalra, P. K. (1998), 'Two new techniques for parameter estimation using neural networks', *The Aeronautical Journal* **102**(2349), 25–30.

Raisinghani, S. C., Ghosh, A. K. and Khubchandani, S. (1998), 'Estimation of aircraft lateral-directional parameters using neural networks', *Journal of Aircraft* **35**(6), 876–881.

Samal, M. K., Anavatti, S. and Garratt, M. (2008), Neural network based system identification for autonomous flight of an eagle helicopter, *in* 'The International Federation of Automatic Control', Proceedings of the 17th World Congress, pp. 7421–7426.

Savran, A., Tasaltin, R. and Becerikli, Y. (2006), 'Intelligent adaptive nonlinear flight control for a high performance aircraft with neural networks', *ISA Transactions* **45**(2), 225–247.

Singh, B. (2005), Online aerodynamic parameter estimation for a fault tolerant flight control system, Master's thesis, Nagpur University.

Singh, S. and Ghosh, A. K. (2007), 'Estimation of lateral-directional parameters using neural networks based modified delta method', *The Aeronautical Journal* **111**(3150), 659–667.

Song, S., Yu, Z. and Chen, X. (2005), 'A novel radial basis function neural network for approximation', *International Journal of Information Technology* **11**(9).

Song, Y., Campa, G., Napolitano, M., Seanor, B. and Perhinschi, M. G. (2001), Comparison of on-line parameter estimation techniques within a fault tolerant flight control system, Technical report, School of Aerospace and Mechanical Engineering, Hankuk Aviation University, South Korea.

Songwu, L. and Basar, T. (1998), 'Robust nonlinear system identification using neural network models', *IEEE Transactions on Neural Networks* **9**(3), 1–24.

Stingu, E. and Lewis, F. L. (2008), 'A hardware platform for research in helicopter uav control', *Journal of Intelligent Robotic Systems* **2**.

Suresh, S. and Kannan, N. (2008), 'Direct adaptive neural flight control system for an unstable unmanned aircraft', *Applied Software Computing* **8**, 937–948.

Suresh, S., Kumar, M. V., Omkar, S. N., Mani, V. and Sampath, P. (1995), Neural networks based identification of helicopter dynamics using flight data, *in* 'Proceedings of the 9th International Conference on Neural Information Processing', Vol. 1, pp. 10–14.

Tischler, M. B. (1995), System identification methods for aircraft flight control development and validation, Technical Report NASA-TM-110369, NASA.

Tischler, M. B. and Remple, R. K. (2006), *Aircraft and Rotorcraft System Identification - Engineering Methods with Flight Test Examples*, AIAA Education Series, American Institute of Aeronautics and Astronautics.

Velo, J. G. and Walker, B. K. (1997), 'Aerodynamic parameter estimation for high performance aircraft using extended kalman filtering', *AIAA Journal of Guidance, Control, and Dynamics* **20**(6), 1257–1259.

White, D. A. and Sofge, D. A. (1992), *Handbook of Intelligent Control - Neural, Fuzzy and Adaptive approaches*, Multisciences Press Inc.

Williams, K. W. (2004), A summary of unmanned aircraft accident/incident data: Human factors implications, Technical report, Civil Aerospace Medical Institute - Federal Aviation Administration.

Williams, K. W. (2006), Human factors implications of unamanned aircraft accidents: Flight-control problems, Technical report, Civil Aerospace Medical Institute - Federal Aviation Administration.

Yeager, J. (1998), Implementation and testing of turbulence models for the f18-harv simulation, Technical Report NASA CR-1998-206938, Lockheed Martin Engineering and Sciences.

Appendix A: State Space Representation for Stability Analysis

Hover Flight

$$A = \begin{bmatrix} 0.0311 & 0 & 0 & 0 & -9.81 & 0 & 0 & 0 & 0 & -9.8091 & 0 \\ 0 & 0.0916 & 0.0009 & 9.8091 & 0 & 0 & 0.0019 & 0 & -0.0211 & 0 & 9.8091 \\ 0 & 0 & -0.7751 & -0.1293 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9999 & -0.0132 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0132 & 0.9999 & 0 & 0 \\ 0 & 0.0846 & 0.0106 & 0 & 0 & 0 & 0.0070 & 0 & -0.0769 & 0 & 405.01 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 214.42 & 0 \\ 0 & -0.6186 & 0 & 0 & 0 & 0 & -0.0514 & 0 & 0.56293 & 0 & 0 \\ 0.0025 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -8.35 & 0 \\ 0 & 0.0025 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -8.35 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1707 & 3.2515 & 0 \\ 0 & 0 & -133.7566 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.8276 & 11.8498 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -86.6513 & -42.7716 \\ 35.0700 & 0 & 0 & 0 & 0 \\ 0 & 35.0700 & 0 & 0 & 0 \end{bmatrix}$$

Forward Flight 10 m/s

$$A = \begin{bmatrix}
-0.1508 & 0 & -0.0340 & 0 & -9.7774 & 0 & 0 & 0.8169 & -0.0013 & -9.7580 & 0 \\
-0.0001 & 0.0761 & 0.0018 & 9.7767 & 0.0095 & 0 & -0.8150 & 0 & -10.0072 & 0 & 9.7580 \\
-0.2976 & 0 & -1.5604 & -0.1160 & 0.7986 & 0 & 0.0013 & 10.0167 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -0.0010 & -0.0817 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9999 & -0.0119 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0119 & 1.0033 & 0 & 0 \\
0.0021 & 0.0289 & 0.0195 & 0 & 0 & 0 & 0.0069 & 0.0001 & -0.0263 & 0 & 404.61 \\
-0.0287 & 0 & 0.4161 & 0 & 0 & 0 & 0 & 0.2859 & 0 & 214.13 & 0 \\
0.0114 & -0.2113 & -0.0009 & 0 & 0 & 0 & -0.0506 & -0.0008 & 0.1923 & 0 & 0 \\
0.0025 & 0 & -0.0001 & 0 & 0 & 0 & 0 & -1 & 0 & -8.35 & 0 \\
0 & 0.0023 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -8.35
\end{bmatrix}$$

$$B = \begin{bmatrix}
0 & 0 & -0.9450 & 0 & 0 \\
0 & 0 & 0.1645 & 3.0690 & 0 \\
0 & 0 & -144.3356 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1.7687 & 11.1847 & 0 \\
0 & 0 & 1.2242 & 0 & 0 \\
0 & 0 & 0 & -81.7880 & -42.7716 \\
35.0700 & 0 & 0.2915 & 0 & 0 \\
0 & 35.0700 & 0 & 0 & 0
\end{bmatrix}$$

Forward Flight 20 m/s

$$A = \begin{bmatrix}
-0.3007 & 0.0009 & -0.0207 & 0 & -9.2555 & 0 & 0 & 7.0245 & -0.1264 & -8.9200 & 0 \\
-0.0054 & 0.0672 & 0.0042 & 9.2540 & 0.0590 & 0 & -7.0217 & 0.0018 & -20.0019 & 0 & 8.9200 \\
-0.1769 & 0.0009 & -1.9183 & -0.1679 & 3.2508 & 0 & 0.1264 & 20.0074 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -0.0064 & -0.3512 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9999 & -0.0181 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.0192 & 1.0597 & 0 & 0 \\
-0.0179 & 0.0074 & 0.0403 & 0 & 0 & 0 & 0.0101 & 0.0065 & -0.0068 & 0 & 395.49 \\
-0.4864 & -0.0001 & 0.4775 & 0 & 0 & 0 & 0 & 0.1270 & 0 & 209.38 & 0 \\
0.1496 & -0.0543 & -0.0525 & 0 & 0 & 0 & -0.0742 & -0.0478 & 0.0494 & 0 & 0 \\
0.0034 & 0 & 0.0010 & 0 & 0 & 0 & 0 & -1 & 0 & -8.35 & 0 \\
0 & 0.0043 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -8.35
\end{bmatrix}$$

$$B = \begin{bmatrix}
0 & 0 & -3.9029 & 0 & 0 \\
0 & 0 & 0.1994 & 3.1358 & 0 \\
0 & 0 & -174.7897 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.9999 & 11.4282 & 0 \\
0 & 0 & 27.0623 & 0 & 0 \\
0 & 0 & 0 & -83.5683 & -42.7716 \\
35.0700 & 0 & 0.6248 & 0 & 0 \\
0 & 35.0700 & -0.0039 & 0 & 0
\end{bmatrix}$$

Appendix B: Parameter Statistics

Mean and Confidence Interval

The mean value can be regarded as the 'central tendency' of a variable. This is calculated as:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

where n is the number of observed values of x. But this method can only be made meaningful with its associated confidence intervals, which give a range of values (upper and lower bounds) if a level of certainty $p = 0.05$. Then there is a 95-percent probability the population mean will fall within these bounds. Given a random sample with a standard deviation σ, the confidence intervals for the population mean are defined as:

$$z_{1,2} = \bar{x} \pm 1.96\sigma/\sqrt{n} \tag{2}$$

The computation of the intervals assume a normal distribution of values around the population mean value. This method indirectly evaluates whether the RBFN has adequately represented the dynamics required to estimate the parameter within the specified confidence level.

Variance and Standard Deviation

The variance of a variable distribution is the expected square value of the deviation of that variable from its expected mean value. In general, the population variance with finite size n is given by:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{3}$$

Standard deviation of a variable distribution is simply the square root of its variance. It shows how much variation there is from the estimated mean value. A low standard deviation indicates the data samples tend to be very close to the expected mean and a higher value indicates the data samples are spread out over a large range of values. Denoted as σ, it is

defined as:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{4}$$

Cramer-Rao Lower Bound

An important question in estimation theory is that the estimated parameter, denoted here as $\hat{\theta}$ converges to the unknown target parameter θ. Unbiasedness is one desirable property that indicates the estimator hits its target given as:

$$E(\hat{\theta}) = \theta \tag{5}$$

In this context, the Cramer-Rao Lower Bound (CRLB) will give the minimal achievable variance for any unbiased estimator. This method is often used to provide a benchmark against which the performance of any unbiased estimator can be compared. The closer the estimate is to the CRLB, the less its variance associated to its mean value. Given a population with mean μ and variance σ^2, the information matrix can be defined:

$$I = \begin{pmatrix} \frac{n}{\sigma^2} & 0 \\ 0 & \frac{n}{2\sigma^4} \end{pmatrix} \tag{6}$$

So given a sample with mean $\bar{x}$ and variance s^2 which are estimators of μ and σ^2 the bound on those estimates is given as:

$$V(\bar{x}) \geq \frac{\sigma^2}{n} \tag{7}$$

and the bound on the best possible estimator of the parameter μ in terms of variance is:

$$V(s^2) \geq \frac{2\sigma^4}{n} \tag{8}$$

These inequalities can also be represented as a percentage away from the optimal estimate with the lowest variance. Given the estimation is performed online, the information on the entire population is not available therefore the CRLB will have little significance on parameter reliability.

Appendix C: Online Estimation Algorithm

The MATLAB source code for the Modified Delta Method and Delta Method online estimation algorithms is described below:

```
% Nonlinear Model identification
% The SIMULINK model needs to be validated based on the mathematical
% equations provided by Gravilet et al

clear all
clc

atmos = 0;
height = 50;
Lw = height/0.3028;
turb = 0;

switch atmos
case 0
sigmaW = 0;
case 1
sigmaW = 0.1*15;
case 2
sigmaW = 0.1*30;
case 3
sigmaW = 0.1*45;
end

% set flight condition
for s = 1:3
cond = s;
% --------------------LINEAR MODEL----------------------
% Initialize linear model
```

```
[A,B,C,D,init,inputs] = InitLinearFunc(s);

% enter control signals: d_lon to d_r
for rs = 4:5
%condition for control
switch rs
case 1
meas_ch = [1 5 7];
case 2
meas_ch = [2 4 8];
case 3
meas_ch = [3 5];
case 4
meas_ch = 6;
case 5
meas_ch = 6;
end

% choice of simulation signal
ch = 1;
% Initialize nonlinear model
[Omeg0,u0,v0,w0,phi0,theta0,psi0,p0,q0,r0,xe0,ye0,ze0,a0,b0,inputs]...
= InitNonlinearFunc(cond);
% set develop identification signal
[d_lon,d_lat,d_col,d_r,d_t,noise,timespan] = SignalGen(rs,ch,inputs,cond);
% yaw gyro noise
% assume no noise for initial comparison
if noise == 0
noisemin = -1e-20;
noisemax = 1e-20;
else
noisemin = -noise;
noisemax = noise;
end

for ch_out = 1:length(meas_ch)

% sliding window parameter
for sldwin = 150
%------running nonlinear simulink model------%
[t,x,y] = sim('OnlineModel',timespan,...
[],d_lon,d_lat,d_col,d_r,d_t);
%-----------------------------------%
```

```
% save choice of control signal
switch rs
case 1
savech = 'dlon';
case 2
savech = 'dlat';
case 3
savech = 'dcol';
case 4
savech = 'dr';
case 5
savech = 'dt';
end
% save choice of flight condition
switch s
case 1
fold = 'Online Estimation\ParaEst\MDM\Hover\';
case 2
fold = 'Online Estimation\ParaEst\MDM\Forward10\';
case 3
fold = 'Online Estimation\ParaEst\MDM\Forward20\';
end

% create counter
counter = length(t);
window_size = size(dlon,2);

tic

for cd =1:counter

input_set = 0;
input_set = [u(cd,:);v(cd,:);w(cd,:);p(cd,:);q(cd,:);r(cd,:);...
phi(cd,:);theta(cd,:);a1(cd,:);b1(cd,:);...
dlon(cd,:);dlat(cd,:);dcol(cd,:);dr(cd,:);dt(cd,:)];

switch meas_ch(ch_out)
case 1
    output_set = diff(ax(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta X';
    savevar = 'X';
```

```
    par_ch = {'u';'theta';'a1'};
    par_valplot = {'{u}';'{\theta}';'{a1}'};
    par_tar = [A(3,3) A(3,7) A(3,15)];
case 2
    output_set = diff(ay(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta Y';
    savevar = 'Y';
    par_ch = {'v';'phi';'b1'};
    par_valplot = {'{v}';'{\phi}';'{b1}'};
    par_tar = [A(4,4) A(4,6) A(4,16)];
case 3
    output_set = diff(az(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta Z';
    savevar = 'Z';
    par_ch = {'w';'d_col'};
    par_valplot = {'{w}';'{\delta_{col}}'};
    par_tar = [A(5,5) B(5,3)];
case 4
    output_set = diff(aL(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta L';
    savevar = 'L';
    par_ch = {'v';'b1'};
    par_valplot = {'{v}';'{b1}'};
    par_tar = [A(9,4) A(9,16)];
case 5
    output_set = diff(aM(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta M';
    savevar = 'M';
    if (cond > 1) & (rs == 3)
        par_ch = {'d_col'};
        par_valplot = {'{\delta_{col}}'};
        par_tar = [B(10,3)];
    elseif rs == 1
        par_ch = {'a1'};
        par_valplot = {'{a1}'};
        par_tar = [A(10,15)];
    end
case 6
    output_set = diff(aN(cd,:));
```

```
    goal = 1e-5;
    plotdisp = '\Delta N';
    savevar = 'N';
    if rs == 4
        par_ch = {'r';'d_r'};
        par_valplot = {'{r}';'{\delta_{r}}'};
        par_tar = [A(11,3) B(11,4)];
    elseif rs == 5
        par_ch = {'d_t'};
        par_valplot = {'{\delta_{t}}'};
        par_tar = [B(11,5)];
    end
case 7
    output_set = diff(acc_a1(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta A';
    savevar = 'A';
    par_ch = {'a1';'d_lon'};
    par_valplot = {'{a1}';'{\delta_{lon}}'};
    par_tar = [A(15,15) B(15,1)];
case 8
    output_set = diff(acc_b1(cd,:));
    goal = 1e-7;
    plotdisp = '\Delta B';
    savevar = 'B';
    par_ch = {'b1';'d_lat'};
    par_valplot = {'{b1}';'{\delta_{lat}}'};
    par_tar = [A(16,16) B(16,2)];
end


% label signals
signals = {'u';'v';'w';'p';'q';'r';'phi';'theta';'a1';'b1';...
'd_lon';'d_lat';'d_col';'d_r';'d_t'};

% setup of input set differential variations
input_set = diff(input_set')';

% compute the standard deviation and mean value of each input signal
Idev = std(input_set');
Imean = mean(input_set');
[rf,s] = size(input_set);
```

```
% initialize counter
j = 0;
count = 0;
for j = 1:rf
if ((Idev(j) < 1e-6) & (Imean(j) < 1e-7))
count(j) = 0;
else
count(j) = 1;
end
end
% find column where
meas = find(count == 0);
% eliminate signals with zero effect
input_set(meas,:) = [];
Idev(meas) = [];
Imean(meas) = [];
signals(meas) = [];
% generate estimation input/output set
est_sig = input_set;
est_out = output_set;

if cd == 1 % guaranteed at time = 0
netsize(cd) = 0;
for i = 1:j
    par_est(i,cd) = 0;
    lower_est(i,cd) = 0;
    upper_est(i,cd) = 0;
    outliers(i,cd) = 0;
end

elseif netsize(cd-1) == 0 % no network validation for size == 0
if isempty(Idev) == 0 % IFF control signal exist then proceed
spread = 0.8326;
net = newrbf(est_sig,est_out,goal,spread); % create network
nnsize = net.layers{1}.size;
nninput = net.inputs{1}.size;
netsize(cd) = nnsize;
netinput(cd) = nninput;
% modified delta method
[j,k] = size(est_sig);
for i = 1:j
    variable = zeros(j,k);
    variable(i,:) = est_sig(i,:);
```

```
    % simulate modified input file
    mdm_out = sim(net,variable);
    denom = est_sig(i,:);
    % find indices that will lead to large variances
    indices = find(abs(denom) < 1e-5);
    denom(indices) = 0;
    result = mdm_out./denom;
    ind = isinf(result);
    result(ind) = [];
    % removal of outliers - 2 iterations
    for h = 1:2
        diffd = result - mean(result);
        ind2 = find(abs(diffd) > std(result));
        result(ind2) = [];
    end
    store(i) = {result};
    par_est(i,cd) = mean(result);
    % 95-percent probability confidence level
    lower_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    upper_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    % this represents the number of outliers removed. The higher the value,
    % the more reliable is the estimate
    outliers(i,cd) = k - length(result);
end
else
netsize(cd) = 0;
netinput(cd) =15-length(meas);
[j,k] = size(est_sig);
% parameter remained unchanged
for i = 1:j
    par_est(i,cd) = par_est(i,cd-1);
    lower_est(i,cd) = lower_est(i,cd-1);
    upper_est(i,cd) = upper_est(i,cd-1);
    outliers(i,cd) = outliers(i,cd-1);
end
end
elseif nninput == 15-length(meas) % validation - compare input sizes
val_out = sim(net,input_set);
% compute the mean square error
error = output_set - val_out;
err_mse = sum(error.^2)/length(error);
if err_mse > 1e-5 % create new network is error is above threshold
spread = 0.8326;
```

```
net = newrbf(est_sig,est_out,goal,spread);
nnsize = net.layers{1}.size;
nninput = net.inputs{1}.size;
netsize(cd) = nnsize;
netinput(cd) = nninput;
% modified delta method
[j,k] = size(est_sig);
for i = 1:j
    variable = zeros(j,k);
    variable(i,:) = est_sig(i,:);
    % simulate modified input file
    mdm_out = sim(net,variable);
    denom = est_sig(i,:);
    % find indices that will lead to large variances
    indices = find(abs(denom) < 1e-5);
    denom(indices) = 0;
    result = mdm_out./denom;
    ind = isinf(result);
    result(ind) = [];
    % removal of outliers - 1 iterations
    for h = 1:2
        diffd = result - mean(result);
        ind2 = find(abs(diffd) > std(result));
        result(ind2) = [];
    end
    store(i) = {result};
    par_est(i,cd) = mean(result);
    % 95-percent probability confidence level
    lower_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    upper_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    % this represents the number of outliers removed. The higher the value,
    % the more reliable is the estimate
    outliers(i,cd) = k - length(result);
end
else
netsize(cd) = netsize(cd-1);
netinput(cd) = netinput(cd-1);
[j,k] = size(est_sig);
% parameter remained unchanged
for i = 1:j
    par_est(i,cd) = par_est(i,cd-1);
    lower_est(i,cd) = lower_est(i,cd-1);
    upper_est(i,cd) = upper_est(i,cd-1);
```

```
    outliers(i,cd) = outliers(i,cd-1);
end
end
else
if isempty(Idev) == 0 % IFF control signal exist then proceed
spread = 0.8326;
net = newrbf(est_sig,est_out,goal,spread); % create network
nnsize = net.layers{1}.size;
nninput = net.inputs{1}.size;
netsize(cd) = nnsize;
netinput(cd) = nninput;
% modified delta method
[j,k] = size(est_sig);
for i = 1:j
    variable = zeros(j,k);
    variable(i,:) = est_sig(i,:);
    % simulate modified input file
    mdm_out = sim(net,variable);
    denom = est_sig(i,:);
    % find indices that will lead to large variances
    indices = find(abs(denom) < 1e-5);
    denom(indices) = 0;
    result = mdm_out./denom;
    ind = isinf(result);
    result(ind) = [];
    % removal of outliers - 1 iterations
    for h = 1:2
        diffd = result - mean(result);
        ind2 = find(abs(diffd) > std(result));
        result(ind2) = [];
    end
    store(i) = {result};
    par_est(i,cd) = mean(result);
    % 95-percent probability confidence level
    lower_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    upper_est(i,cd) = 1.96*std(result)/sqrt(length(result));
    % this represents the number of outliers removed. The higher the value,
    % the more reliable is the estimate
    outliers(i,cd) = k - length(result);
end
else
netsize(cd) = 0;
netinput(cd) =15-length(meas);
```

```
[j,k] = size(est_sig);
% parameter remained unchanged
for i = 1:j
    par_est(i,cd) = par_est(i,cd-1);
    lower_est(i,cd) = lower_est(i,cd-1);
    upper_est(i,cd) = upper_est(i,cd-1);
    outliers(i,cd) = outliers(i,cd-1);
end
end
end % IF loop
end %FOR loop network computation - per time window

[j,k] = size(est_sig);
for i = 1:length(par_ch)
for ii = 1:j
check(ii) = isequal(par_ch(i),signals(ii));
if check(ii) == 1
channel(i) = ii; % pre-determined channels
end
end
end

% PARAMETER PLOTTING SECTION
% --------------------------

for i= 1:length(par_ch)
par_plot = par_est(channel(i),:);
boundary = lower_est(channel(i),:);
% choose every 20 value
val_count = 1:20:length(par_est);
par_plot = par_plot(val_count);
boundary = boundary(val_count);
par_time = t(val_count);
errorbar(par_time,par_plot,boundary,'o:','Linewidth',2);
grid on;
hold on;
plot(t,par_tar(i).*ones(1,length(t)),'k','Linewidth',2);
legend('NN output','linear model');
xlabel('time \it sec');
ylabel(strcat(savevar,'_',par_valplot(i)));
set(gcf,'PaperUnits','points','PaperPosition',[40,200,500,400]);
set(gcf,'PaperType','A4');
whitebg([1 1 1]);
```

```
set(gcf,'Color',[1,1,1]);
saveas(gcf,[fold,savevar,par_ch{i},'noise',num2str(noise*100),'.pdf']);
cla;
sav_plot = par_est(channel(i),:);
save([fold,savevar,par_ch{i},'noise',...
num2str(noise*100),'stack',...
num2str(sldwin),'.mat'],'sav_plot');
end
end % FOR loop sliding window choice
end % FOR loop measure output choice
end % FOR loop control signal choice
end % FOR loop flight condition choice
```