1003_F2

by Ankit Gupta

FILE	1003_F2.PDF (2.77M)		
TIME SUBMITTED	29-MAR-2017 04:55PM	WORD COUNT	22766
SUBMISSION ID	791210125	CHARACTER COUNT	111054

AE3212-II Simulation,Verification & Validation

Flight Dynamics

7th-29th March 2017

Group 1003F2

Sumalik Boddu	4381866
Alexander Gilbert	4352246
Thomas Govaert	4345436
Bas Grootnibbelink	4357132
Kiko Guimaraes	4343727
Ankit Gupta	4355458

TU Delft

Contents

1 Introduction 1

2 Problem Analysis 1
- 2.1 Problem Statement . . . 2
- 2.2 Input & Output Parameters . . . 2
- 2.3 Reference Frames . . . 3
- 2.4 General Assumptions . . . 3
- 2.5 Governing Equations . . . 5

3 Experimental Data Analysis 8
- 3.1 Mass & Balance . . . 8
- 3.2 Stationary Measurements . . . 8
- 3.3 Eigenmotion Measurements . . . 11
- 3.4 Missing Data . . . 12

4 Analytical Model 13
- 4.1 Symmetric Equations of Motion . . . 13
- 4.2 Asymmetric Equations of Motion . . . 14
- 4.3 Analytical Results . . . 15
- 4.4 Effects of Assumptions . . . 16

5 Numerical Model 16
- 5.1 State-Space Form of Symmetric Equations . . . 16
- 5.2 State-Space Form of Asymmetric Equations . . . 17
- 5.3 Eigenvalues . . . 18

6 Computational Model 19
- 6.1 Numerical Algorithm . . . 19
- 6.2 Flowchart . . . 20

7 Verification 20
- 7.1 Code Verification . . . 20
- 7.2 Unit Testing . . . 20
- 7.3 System Testing . . . 22

8 Validation 23
- 8.1 Validation Experiment . . . 23
- 8.2 Comparison of Eigenvalues and Errors . . . 23
- 8.3 Simulated Responses . . . 24
- 8.4 Uncertainty Assessment . . . 27
- 8.5 Key Stability Derivatives Tweaking . . . 29
- 8.6 Recommendations . . . 30

9 Conclusion 32

Bibliography 32

A Task Distribution 33

B Simulation Code 34

Chapter 1

Introduction

After the completion of the initial conceptual design of an aircraft, the preliminary design phase begins, whereby it is the job of engineers to analyse and design specialist areas of the aircraft [1]. This project team of control engineers is tasked with such a role: an investigation into the stability and control properties of a new business jet. The dynamic stability behaviour of the jet is instrumental in characterising the aircraft's temporal response to control inputs and disturbances such as turbulence [2]. A computational model, or 'simulation', of the jet's stability was built, consisting of modules of varying degrees of freedom which examine its characteristic eigenmotions. The results of this simulation are to be used in further development of the jet, as it approaches the latter stages of its design. Therefore, it was essential to adequately verify and validate the simulation, otherwise intractable and prohibitively expensive design alterations may have arisen later. Consequently, the team's work is of utmost importance to the manufacturer; not only for the development of a safe aircraft, but for one that will also be a commercial success.

The problem can be generally articulated as: "What is the aircraft's response to control and disturbance inputs?". For building a simulator that could solve this problem, the team began by transitioning from the 'reality of interest' of the problem towards a mathematical model, composed of analytical and numerical solutions [3]. This necessitated the use of simplifying assumptions to facilitate the derivation of equations to describe the aircraft's motion in flight. Moreover, the process involved the use of experimental data from a prototype flight test to calculate relevant parameters in these equations of motion. The computational model was created by implementing the numerical model algorithmically in the Python programming language. A modified 'waterfall model' was employed for devising this program [4]. This included a continual verification process over the duration of the project, whereby unit testing of code was executed from the start; integration testing was carried out on grouped units; system testing was performed once the full simulation was complete. Finally, the recorded input variables from the flight test were fed into the computational model. These simulated results were compared to the actual experimental results, thereby providing a basis for validation of the simulation. Certain proposed modifications to the computational model were detailed and final conclusions were drawn from the work.

The structure of this report is fashioned in a systematic manner that reflects the sequential, functional flow of the project. The physical problem and the approach to simplifying it are laid out in Chapter 2. Chapter 3 examines the results of the flight test, and details the mass and balance calculations of the aircraft. The analytical model, with its supplementary assumptions, is presented in Chapter 4; results for certain inputs are also given. The numerical model, which only uses the original assumptions made to the problem, is presented in Chapter 5. The algorithmic construction of the numerical model in Python is detailed in Chapter 6; the simulation flow chart is also found in this chapter. The continual verification process of the simulation code is detailed in Chapter 7; furthermore, the verified simulation results, after all unit and system testing was completed, are presented in this chapter. Chapter 8 presents the validation process of the verified results, together with recommendations for future improvement. The report is concluded in Chapter 9, where the findings are summarised and the most remarkable results are appraised. After this final chapter, the interested reader may also read through the bibliography, Appendix A for the team's work division, and Appendix B for the simulation code.

Chapter 2

Problem Analysis

The team was tasked with simulating the response of the aircraft to certain control and disturbance inputs. A problem statement was formulated, Section 2.1. The process of breaking the problem down into something that could be represented conceptually required identifying the input and output variables, presented in Section 2.2. The frames of reference for describing the aircraft's motion were defined Section 2.3. Various assumptions were made to simplify the description of its motion in these reference frames, Section 2.4. Furthermore, the governing equations, for deriving the equations of motion, were identified, Section 2.5.

2.1. Problem Statement

The primary goal of the task at hand is to construct a simulation model that reliably models the dynamic behavior of a new business jet. The model shall be able to accurately predict the static and dynamic stability properties. Thus, the problem can be stated as: "What is the temporal response of the pertinent aircraft states to control and disturbance inputs?"

2.2. Input & Output Parameters

In order to fulfil the primary goal of creating a simulation model for aircraft responses, the required inputs and expected outputs of the model have been determined.

The inputs required for the simulation model are aircraft dimensions, aircraft weights and flight test measurements, shown in Table 2.1, and the aircraft stability derivatives, as shown in Table 2.2 and Table 2.3. It can be noted that these combined make up the problem boundary conditions, the initial conditions and the system properties, enabling problem solving.

Secondly, as mentioned in Section 2.1, it is important to predict the aircraft stability properties. These can be evaluated based on the eigenvalues, since these reflect the system stability. The eigenvalues were determined using the same input parameters as for the simulation model.

1

Table 2.1: Flight Test Parameters

Parameter	Description	Value [Unit]
Aircraft Dimensions		
b	Wing span	15.911 [m]
$\bar{c}$	Mean aerodynamic chord	2.0569 [m]
S	Wing surface area	30.0 [m^2]
Aircraft Weights		
BEW	Basic Empty Weight	9165.0 [lbs]
W_{fuel}	Weight of fuel	2700.0 [lbs]
$W_{payload}$	Weight of payload	762.0 [kg]
W_s	Standard aircraft weigth	60500 [N]
x_{cg}	Centre of gravity	0.25$\bar{c}$ [m]
Flight Test Measurements		
h_p	Pressure altitude	[m]
TAT	Total Air Temperature	[K]
IAS	Indicated airspeed	[m/s]
α	Angle of attack	[°]
ϕ	Roll Angle	[°]
θ	Pitch Angle	[°]
p	Roll Rate	[°/s]
q	Pitch Rate	[°/s]
r	Yaw Rate	[°/s]
ρ_0	Sea level density	[kg/m^3]
T_0	Sea level temperature	[K]
λ_T	Temperature Gradient	[K/m]
R	Specific Gas Constant	[$m^2/(s^2 \cdot K)$]
δ_a	Aileron deflection	[°]
δ_e	Elevator deflection	[°]
δ_r	Rudder deflection	[°]
F_e	Control force	[N]
FFl	Fuel Flow left	[kg/s]
FFr	Fuel Flow right	[kg/s]

2

Table 2.2: Symmetric Stability Derivatives

Longitudinal force derivatives		Normal force derivatives		Pitch moment derivatives	
Coeff.	**Value**	**Coeff.**	**Value**	**Coeff.**	**Value**
C_{X_u}	-0.0279	C_{Z_u}	-0.3762	C_{m_0}	0.0297
C_{X_α}	-0.4797	C_{Z_α}	-5.7434	C_{X_u}	0.0699
$C_{X_{\dot{\alpha}}}$	0.0833	$C_{Z_{\dot{\alpha}}}$	-0.0035	C_{m_α}	-0.5626
C_{X_q}	-0.2817	C_{Z_q}	-5.6629	$C_{m_{\dot{\alpha}}}$	0.1780
C_{X_δ}	-0.0373	C_{Z_δ}	-0.6961	C_{m_q}	-8.7941
				C_{m_δ}	-1.1642
				$C_{m_{T_C}}$	-0.0064

Table 2.3: Asymmetric Stability Derivatives

Lateral force derivatives		Roll moment derivatives		Yaw moment derivatives	
Coeff.	**Value**	**Coeff.**	**Value**	**Coeff.**	**Value**
C_{Y_β}	-0.7500	C_{l_β}	-0.1026	C_{n_β}	0.1348
$C_{Y_{\dot{\beta}}}$	0	C_{l_p}	-0.7108	$C_{n_{\dot{\beta}}}$	0
C_{Y_p}	-0.0304	C_{l_r}	0.2376	C_{n_p}	-0.0602
C_{Y_r}	0.8495	$C_{l_{\delta_a}}$	-0.2309	C_{n_r}	-0.2061
$C_{Y_{\delta_a}}$	-0.0400	$C_{l_{\delta_r}}$	0.0344	$C_{l_{\delta_a}}$	-0.0120
$C_{Y_{\delta_r}}$	0.2300			$C_{l_{\delta_a}}$	-0.0939

Besides inputs, it is also crucial to know which outputs are required. As stated in Section 2.1, both aircraft response and stability are asked for. The response is represented by the state variables for symmetrical and asymmetrical motion as shown in Table 2.4. While the outputs for this stability analysis are the system or aircraft eigenvalues, also presented in Table 2.4. For the definitions of symbols used in this report, the reader

Table 2.4: Simulation Outputs

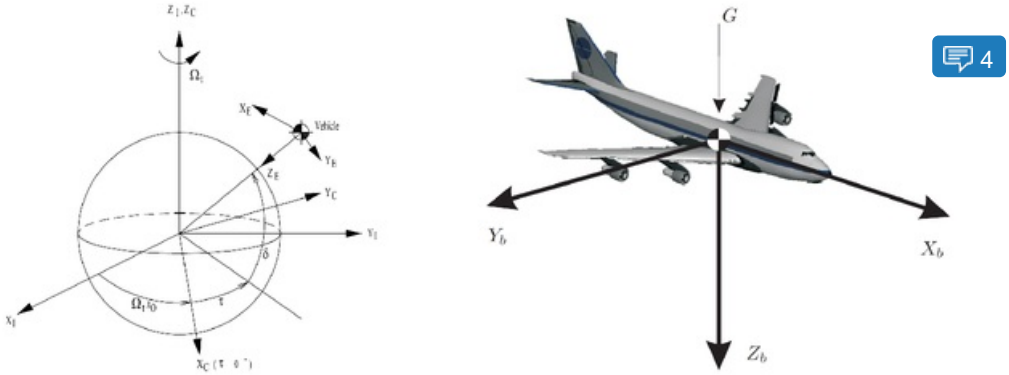
Parameter	Description	Value [Unit]
Symmetric State Variables		
V_t	True Airspeed	m/s
α	Angle of Attack	rad
θ	Pitch Angle	rad
q	Pitch Rate	rad/s
Asymmetric State Variables		
ϕ	Roll Angle	rad
p	Roll Rate	rad/s
r	Yaw Rate	rad/s
System Eigenvalues		
λ	System Eigenvalues	[-]

3

is referred to Appendix A of the AE3212-I Flight Dynamics Lecture Notes [5].

2.3. Reference Frames

A frame of reference is a form of coordinate system that is necessary to gain knowledge about the position and the orientation of an aircraft at any point in time. There are two fundamental reference frames, namely the Earth-Centred Inertial reference frame (ECI; denoted by subscript 'I') and the body-fixed reference frame (denoted by subscript 'b'), that are going to be used throughout this exercise to investigate the forces and moments. They are presented in Figure 2.1a and Figure 2.1b respectively. As sign convention, positive sense of different axes is determined by following the right-handed rule. Figure 2.1a also includes the Earth- Centered Earth Fixed reference frame(ECEF; denoted by subscript 'C') which is basically rotating along with the earth. Furthermore, it also contains the Vehicle Carried Normal Earth reference frame (denoted by subscript 'E').



4

(a) Earth-centered inertial reference frame [6].

(b) Body-fixed reference frame. G is the centre of gravity [6].

Figure 2.1: Reference frames used in derivation of aircraft equations of motion.

2.4. General Assumptions

The flight operating conditions under investigation assume a typical civil flight profile at $M \leqslant 0.8$. Ideally, a further condition would be a cruise altitude of FL410 or higher, which is typical of business jets [7]; however, due to limitations, the flight test could only be performed at a much lower altitude and hence this factor is disregarded. Furthermore, a clean (flaps up; gear up) aircraft configuration was assumed, corresponding to the configuration used in the flight test. These flight conditions provided the setting for deriving the equations of motion. However, the complexity of the problem necessitated the use of various simplifying assumptions, which concern the derivation of easily solvable equations of motion for the business jet; from these, suitable analytical and numerical models could be made.

The assumptions are listed in the bullet points below. They can generally be categorised as being applied to the Earth; to the aircraft; or to forces acting on the aircraft. Note that the expected effects of each assumption on the accuracy of the results is discussed; where possible, these effects are quantified. Such effects pertain to discrepancies between the outputs of the simulation and what is actually recorded in the flight test (modelling errors). The reader is referred to Subsection 8.4.1 in the Validation chapter, where these effects are elaborated upon in relation to their impact on the actual results obtained.

- **Earth is non-rotating**. This fixes the ECEF frame in place, such that it has no rotational motion with respect to ECI, thereby making ECEF an inertial frame as well; hence Newton's laws hold in the frame and the Coriolis and centrifugal accelerations are eliminated. The longer the time span of the motion, the larger the accumulated errors in the aircraft's states will become. For the relatively short durations under investigation, this only introduces small errors [5].

- **Earth is flat**. Curvature of the Earth is neglected; its radius becomes infinite. This results in terms with R in the denominator becoming zero. Moreover, the local horizontal plane, $X_E Y_E$, does not rotate and only translates. Further, if the E-frame is set to have the same orientation as the ECI frame at t=0, both frames have the same orientation with respect to each other for all t; this makes the rotation transformation matrix the identity matrix and thus only a translation is required to transform from one frame to the other. Considering the short distances covered during the eigenmotions being investigated, the assumption generates negligible error due to the relatively gentle curvature of the Earth [5]. The error attributed to the E-frame orientation is estimated at less than 0.36% for the typical spatial ranges of the eigenmotions of interest (refer to Subsection 8.4.1 for further explanation).

- **Earth is spherical**. For purposes of latitude and gravity, this assumption does not conflict with the previous one stating that the Earth is flat. The Earth's flattening and geometric irregularities induce gravitational deviations from the 1st order Kepler model of a spherical, radially symmetric Earth. The effect is twofold. Firstly, the geocentric, gravitation, and geodesic latitudes all become equal. Secondly, the Legendre polynomials are removed from the classical gravitational potential and can simply be described by the Newtonian central field term [8]; the gravity vector thus points directly at the centre of the Earth, rather than off-centre as is the case with an ellipsoid Earth. The most prominent irregularity in the gravitational potential is the J_2 term at the equatorial bulge [8]; as such, moving across a large latitudinal range near the equator would result in the largest errors. Since the flight test is performed in the Netherlands over a very small latitudinal range, it was expected that this error is negligible.

- **Gravity force is constant**. With the previous assumption that the Earth is spherical, the zonal, sectorial, and tesseral correction terms in the gravity potential are removed and only the 1st order Kepler term remains [8]. This allows further simplification by assuming that this term is constant with radial distance from the sphere's centroid. Thus, gravitational acceleration is set to sea-level (g = 9.81 m/s^2), thereby ignoring altitude effects. The effect on accuracy is small (at 32 km altitude, there is a 1% error [5]).

- **Wind force is zero**. This implies that the unperturbed air is motionless with respect to the Earth's surface and effects of wind shear and turbulence are hence neglected. Consequently, the kinematic velocity is equal to the aerodynamic velocity since the aerodynamic force acting on the aircraft is directly expressed in the kinematic frame without transformation [5]. For typical flight conditions, the magnitude of the kinematic velocity is much greater than that of the wind. However, for lower flight speeds, stronger wind shears, or storm conditions, larger errors can be expected.

- **Aircraft is a rigid body of constant mass**. A rigid body is non-elastic. Coupled with the assumption that mass is constant (no fuel consumption) and mass distribution is invariant (no passenger, crew, or landing gear movement), the Coriolis and relative forces expressed in the inertial frame are zero; this results in a constant inertia matrix [5]. This is attributed to the 'Principle of Solidification', which states that the equations of motion for an elastic, mass-varying body are equivalent to those for a rigid body plus the effect of Coriolis and relative forces and moments [9]. As a result, the error is dependent on the magnitude of the neglected apparent forces and moments with respect to the true forces and moments.

- **Aircraft does not possess rotating masses**. Turbine blades in the jet engines are assumed non-rotating; gyroscopic forces are thus neglected [10]. Considering the relatively small mass of the blades and small diameter of the engines, this effect is not significant and can safely be ignored.

- **Aircraft I_{xy} and I_{yz} equal zero**. The business jet has a typical configuration and its design is not expected to deviate to something more exotic during its future development. As such, X_bZ_b can be considered a plane of symmetry which causes the rotational inertia tensors, I_{xy} and I_{yz}, to become zero. The error that arises from this assumption lies with the degree to which the aircraft is in fact mass-asymmetric around the X_bZ_b plane. Imperfect fuel distribution in the wings and payload distribution leads to this error. With typical loading procedures, the error can be neglected.

- **Aircraft is in horizontal, steady, symmetric flight at t<0**. In the instant before the control input is made, the aircraft is flying a typical cruise. This assumption has implications for the initial conditions of the equations of motion, setting initial rotational rates to zero. In reality, however, it is expected that there will always be minute deviations from the perfect horizontal, steady, symmetric flight.

The equations of motion that are derived using the aforementioned assumptions are non-linear. For this investigation, study of the characteristic modes of the aircraft is the objective; therefore, the equations must be linearised [6]. To this end, further assumptions were made, as listed below.

- **Aircraft steady state is statically stable**. This means that the aircraft will not experience deviations that quickly move it away from its equilibrium point; it will always return to the equilibrium in theory, even if it is dynamically unstable. Linearisation cannot accurately describe the motion in this case [5].

- **Aircraft does not deviate significantly from the steady state flight condition**. Linearisation is accurate for the selected flight condition, as well as for conditions near this steady state condition [5]. Deviation far from this point results in significant errors.

For the analysis of both stationary measurement series of flight data, a distinct set of assumptions was used, common to both series; these are listed below.

- **Air is a calorically perfect, ideal gas**. The specific heats of air, c_p and c_v, become constant; hence, the specific heat ratio is constant and does not vary with temperature [11].

- **The wing's C_N equals C_L**. This sets the normal force coefficient equal to the lift coefficient. For the range of angle of attack investigated during the stationary measurements, the error is very small [5].

- **Aircraft engines are at the same incidence as the wing**. This sets $\alpha_T = 0$, simplifying the analysis. This thrust angle of attack is expected to remain very small for the foreseeable development of the jet; therefore, error is expected to be negligible.

Further, for the first stationary series, a supplementary assumption was made, listed below.

- **Aircraft is in horizontal, steady, symmetric flight**. For the duration of the stationary measurements, this implies $\frac{dV}{dt} = 0$, $\frac{d\gamma}{dt} = 0$, $\gamma = 0$, and altitude is constant [12]. In reality, there are always perturbations present; for purposes of this particular flight test, these could be assumed negligible.

For the second stationary series, another set of supplementary assumptions were made, listed below.

- **Aircraft is in straight, quasi-steady, symmetric flight**. For the duration of the stationary measurements, this implies $\frac{dV}{dt} = 0$, $\frac{d\gamma}{dt} = 0$, γ is constant, and altitude is constant [12]. In reality, there are always perturbations present; for purposes of this particular flight test, these could be assumed minor.

- **The flight path angle is small**. This allows the small angle approximation to be made, thereby setting $\cos(\gamma) \approx 1$ whilst $\sin(\gamma) \neq 0$. For the test, the highest γ was 18.33°. Thus, $\cos(\gamma) = 0.949$ in reality, yielding an error of 5.35%. This error is fairly small, but not negligible.

2.5. Governing Equations

This section pertains to describing all the governing equations that are used to construct the experimental. analytical and numerical models; they are categorised into different segments and are discussed.

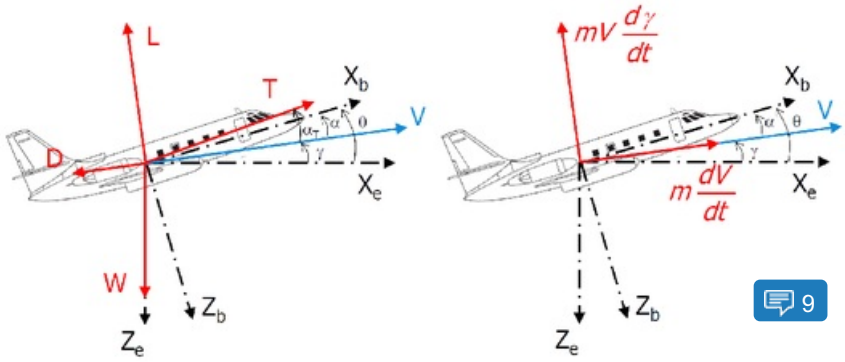


Figure 2.2: Free Body Diagram (left) and Kinetic Diagram (right) [13].

2.5.1. Equations of Motion

To process the data sets of the stationary measurement series, it is necessary to first derive the equations of motion for the corresponding horizontal, stationary and symmetric flight. The configuration analysed is clean i.e., with landing gears retracted and flaps up. To better visualise the forces acting on the aircraft, a free body diagram and a kinetic diagram are presented in Figure 2.2. The basic static equilibrium equations are then applied, Equation 2.1 and Equation 2.2.

$$\sum F_{||V} : \frac{W}{g}\frac{dV}{dt} = T\cos(\alpha_T) - D - W\sin(\gamma) \tag{2.1}$$

$$\sum F_{\perp\_V} : \frac{W}{g}V\frac{d\gamma}{dt} = L - W\cos(\gamma) + T\sin(\alpha_T) \tag{2.2}$$

For steady flight, $\frac{dV}{dt} = 0$, and for horizontal flight, $\frac{d\gamma}{dt} = 0$ and $\gamma = 0$. In addition, it is assumed that thrust angle of attack is zero [12]. Thus, Equation 2.1 and Equation 2.2 simplify to Equation 2.3.

$$\sum F_{||V} : T = D \qquad ; \qquad \sum F_{\perp\_V} : L = W \tag{2.3}$$

2.5.2. Aerodynamic Properties

Now that all the forces acting on the aircraft are known, the lift and drag coefficients can be readily obtained using Equation 2.4. As can be seen, there are some supplementary variables that are required to calculate C_L and C_D.

$$L = C_L\frac{1}{2}\rho V^2 S \rightarrow C_L = \frac{W}{\frac{1}{2}\rho V^2 S} \qquad ; \qquad D = C_D\frac{1}{2}\rho V^2 S \rightarrow C_D = \frac{T}{\frac{1}{2}\rho V^2 S} \tag{2.4}$$

To begin with, the weight of the aircraft changes as a function of time due to constant burning of fuel. Thus, the instantaneous weight or the total weight of the aircraft at a specific point in flight can be formulated by Equation 2.5. The second term on the right hand side of the equation is indicated by the Flight Test Instrument System (FTIS).

$$W(t) = \mathrm{RW} - \int_0^t \dot{m}_{f_{l+r}}\, dt \tag{2.5}$$

The measurements acquired during the test flight are often influenced by uncontrollable variations in atmospheric conditions and in aircraft and engine settings. Therefore, certain corrections for the effected variables have to be made to account for them. This also helps to better predict and compare the obtained results.

$$p = p_0\left(1 + \frac{\lambda h_p}{T_0}\right)^{-\frac{g_0}{\lambda R}} \tag{2.6}$$

$$M = \sqrt{\frac{2}{\gamma - 1}\left[\left(1 + \frac{p_0}{p}\left\{\left(1 + \frac{\gamma - 1}{2\gamma}\frac{\rho_0}{p_0}V_c^2\right)^{\frac{\gamma}{\gamma-1}} - 1\right\}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]} \tag{2.7}$$

Firstly, the airspeed displayed on the pilot screen, i.e., calibrated air speed (V_c) is reduced to equivalent airspeed (V_e). This includes a series of steps, involving Mach number and true airspeed (V_t). The mach number follows directly from Equation 2.7 and the required static pressure (p) is calculated using Equation 2.6. Here, λ is the temperature gradient, R is the specific gas constant of air, γ is the air ratio of specific heats, and p0 and ρ_0 are the ISA values for sea-level static air pressure and air density, respectively.

$$T = \frac{T_m}{1 + \frac{\gamma-1}{2} M^2} \quad ; \quad a = \sqrt{\gamma R T} \tag{2.8}$$

Secondly, the measured total air temperature is corrected for ram rise and this is then followed by evaluating the speed of sound. This is done using Equation 2.8

$$V_t = M \cdot a \quad ; \quad V_e = V_t \sqrt{\frac{\rho}{\rho_0}} \quad ; \quad \tilde{V}_e = V_e \sqrt{\frac{W_s}{W}} \tag{2.9}$$

Finally, from Equation 2.9, the true airspeed and the corresponding equivalent airspeed for that flight condition is obtained. The air density can be easily calculated using ideal gas law. In addition, the aircraft weight is also influenced during the test and this effect is incorporated in reduction of equivalent airspeed by correcting for constant standard weight (W_s = 60500 N).

$$Th = f(h_p, M, \Delta T, \dot{m}_f) \quad ; \quad \Delta T = T - T_{ISA} \quad ; \quad T_{ISA} = T_0 + \lambda h_p \tag{2.10}$$

$$T_c = \frac{Th}{\rho V_t^2 D^2} \quad ; \quad T_{c_s} = \frac{Th_s}{\rho_0 V_e^2 D^2} \tag{2.11}$$

Moreover, the thrust of the engine (Th) is required for the calculation of the drag coefficient. It mainly depends on three parameters- engine control setting (Γ), Mach number and the altitude for a given aircraft configuration in the standard atmosphere. The engine manufacturer has provided a so-called "Computer Card Deck" which uses fuel flow as the engine setting. Using these parameters as inputs, a computer program called thrust.exe was developed that determines the thrust per engine. In order to account for non-standard conditions, it also requires temperature correction (Δ T), which is the difference between the actual outside air temperature (T) and the temperature in the standard atmosphere (T_{ISA}), see Equation 2.10. The dimensionless thrust and standard thrust coefficients then follow from Equation 2.11. For the later, the standard mass flow ($\dot{m}_{f_s}$ = 0.048 kg/s) is used. Here, D is the diameter of the engine, provided by the engine manufacturer [3].

$$Re = \frac{\rho V \bar{c}}{\mu} \quad ; \quad \mu = \frac{\beta \cdot T^{\frac{3}{2}}}{T + S} \tag{2.12}$$

Furthermore, to gain more insight into the airflow patterns, the range of Reynold's numbers, in which the aircraft is being operated, is determined using Equation 2.12 [11]. Here, μ is the dynamic viscosity. For air, $\beta = 1.458.10^{-6}$ and S=110.4 K.

2.5.3. Longitudinal Stability Derivatives

There are two major longitudinal stability derivatives - elevator effectiveness, C_{m_δ} and longitudinal stability, C_{m_α}. The former one is obtained by conducting a small experiment during the flight test. It is known that, for a change in moment, a certain elevator deflection ($\Delta\delta_e$) must be applied in order to return to equilibrium. This change was produced by shifting the centre of gravity to another location. The shift (Δx_{cg}) was achieved by replacing Observer3 from its seat to near the cockpit cabin. Consequently, the elevator effectiveness and longitudinal stability are calculated using Equation 2.13

$$C_{m_\delta} = -\frac{1}{\Delta\delta_e} C_N \frac{\Delta x_{cg}}{\bar{c}} \quad ; \quad C_{m_\alpha} = -C_{m_\delta} \frac{d\delta_e}{d\alpha} \tag{2.13}$$

Once again, noticing that the measurements are conducted under non-standard conditions, the elevator deflection angle is converted to its reduced value. This is done using Equation 2.14. Here, $C_{m_{TC}}$ is the dimensionless thrust moment arm and the computation of T_c and T_{c_s} is dealt in Subsection 2.5.2

$$\delta^*_{e_{eq}} = \delta_{e_{eqmeas}} - \frac{1}{C_{m_\delta}} C_{m_{TC}} (T_{C_s} - T_C) \tag{2.14}$$

$$F_{e_{meas}} = F_{e_{aer}} + F_{e_f} \quad ; \quad F_e^* = F_{e_{aer}} \frac{W_s}{W} \tag{2.15}$$

Similarly, the control force is also effected and thus, need to be reduced to standard conditions. The control force is divided into three parts as can be viewed from Equation 2.15. Here, $F_{e_{meas}}$ is the measured elevator control force, $F_{e_{aer}}$ is the force required to counteract the aerodynamic hinge moment and F_{e_f} is the force required to overcome friction in the control mechanism. Aircraft manufacturers usually manage to minimize the friction as much as possible and besides, any airframe vibrations during flight have a dithering effect and thus, will further reduce friction. Due to these reasons, the second term can be neglected. On incorporating the non-standard aircraft mass effect, the expression for reduced control force can be found.

Chapter 3

Experimental Data Analysis

Experimental results are required for two purposes: building models and, subsequently, validating the simulation [3]. To this end, a test flight was performed in a prototype aircraft. A multitude of parameters, required for the analytical and numerical model designs, were ascertained through this test. Mass and balance computations were carried out post-flight, Section 3.1. In-flight, a set of stationary measurements were made, Section 3.2; this was followed by demonstration of the aircraft's eigenmotions, which were recorded by the computer system. The outputs and state responses were subsequently used for validating the computational model, a process which is elucidated later in Chapter 8.

3.1. Mass & Balance

The mass, W (hereafter referred to as 'weight', per standard aviation convention), and centre of gravity, x_{cg}, of the aircraft are essential parameters for determining the aircraft's stability. These parameters, as functions of time, were determined using the mass and balance manual of the prototype aircraft. Each of the aircraft's weight components are assigned a fuselage station, which lies a fixed distance from an arbitrarily chosen datum line; these moment arms result in a datum-referenced centre of gravity, $x_{cg,datum}$. Taking the sum total of these moments and dividing by the total aircraft weight provides a means to determining x_{cg}.

The Basic Empty Weight (BEW) was first determined from a prior scale weighing. Note that this weight is assumed equal to the Operational Empty Weight (OEW), since operational items weight was assumed negligible [14]. Hence, the Zero Fuel Weight (ZFW) is equal to BEW plus the payload; Ramp Weight (RW) is equal to ZFW plus the block fuel. In this case, the payload only consisted of the pilots and team members. Table 3.1 shows the payload computations, whilst Table 3.2 shows the completed mass and balance form [15].

The $x_{cg,datum}$ of 7.135 m at RW is converted with respect to the forward end of the mean aerodynamic chord, which lies 6.641 m from the datum; this value, x_{cg}, equals 0.4938 m. Having assessed the RW and the corresponding x_{cg}, $W(t)$ and $x_{cg}(t)$ could be evaluated based on the fuel flow measured at various instances during flight. Equation 3.1 gives the weight of the aircraft as a function of time, where the integral indicates the area underneath the curve of combined fuel flow (left and right engines) against flight time; or, in other words, the weight of fuel burnt.

$$W(t) = \mathrm{RW} - \int_0^t \dot{m}_{f_{l+r}}\, dt \tag{3.1}$$

The weight, at a certain measurement time, was thus calculated by subtracting burnt fuel from RW. A form containing fuel moments with respect to the datum line lists the moment arm for 100 lb increments of fuel mass [15]. The relationship is linear; hence, each fuel mass moment arm, at a certain measurement time, was calculated by linear interpolation. The momentary weight, and corresponding shifted x_{cg}, at each measurement time are listed in Table 3.3.

11

3.2. Stationary Measurements

At the time of flight test, two sets of stationary measurements were taken that allows to derive various aerodynamic properties that, at a later stage, aid in refining the simulation model. The C_L-C_D curves follow from the first measurement series as described in Subsection 3.2.1. Through this analysis, the lift-curve slope, the

Table 3.1: Payload mass computation.

Seat	$x_{cg,datum}$ [m]	W [kg]	Moment [kgm]
Pilot 1	3.327	99	329.37
Pilot 2	3.327	90	299.43
Coordinator	4.318	99	427.48
Observer 1L	5.436	74	402.26
Observer 1R	5.436	66	358.78
Observer 2L	6.375	79	503.63
Observer 2R	6.375	80	510
Observer 3L	7.315	91	665.67
Observer 3R	7.315	84	614.46
Baggage	$x_{cg,datum}$ [m]	**W [kg]**	**Moment [kgm]**
Nose	1.880	0	0
Aft Cabin 1	8.153	0	0
Aft Cabin 2	8.585	0	0
Payload	-	762	4,111.20

Table 3.2: Mass and balance computation. Note that the bold lines indicate that the mass below is the sum of the preceding two masses.

	W [kg]	Moment [kgm]
BEW	4,157.17	30,852.14
$x_{cg,datum}$ at BEW = 7.421 m		
Payload	762	4,111.20
ZFW	4,919.17	34,963.33
$x_{cg,datum}$ at ZFW = 7.108 m		
Fuel	1,224.70	8,870.90
Ramp Weight	6,143.87	43,834.23
$x_{cg,datum}$ at RW = 7.135 m		

Table 3.3: Total weight and location of center of gravity during stationary measurements

Time	W [kg]	x_{cg} [m]
Ramp		
00:00:00	6,143.87	0.4938
First Series		
00:17:09	5,998.72	0.4918
00:18:38	5,986.02	0.4916
00:20:28	5,971.05	0.4914
00:21:47	5,963.80	0.4913
00:23:14	5,954.73	0.4911
00:25:05	5,943.84	0.4910
Second Series		
00:29:40	5,900.29	0.4903
00:30:42	5,893.49	0.4902
00:31:48	5,885.33	0.4900
00:32:56	5,879.43	0.4899
00:34:37	5,866.27	0.4897
00:36:45	5,850.40	0.4895
00:37:48	5,843.14	0.4894
C.G. Shift		
00:39:33 (before)	5,832.26	0.4892
00:41:14 (after)	5,819.55	0.4267

zero-lift drag coefficient and the Oswald factor are determined. The second measurement series, as presented in Subsection 3.2.2, is used to calculate the stability properties of the aircraft. Moreover, the reduced elevator trim and reduced elevator control force curves are examined.

3.2.1. First Series

The data acquired from this measurement set is used to compute several other parameters that serve as inputs to constructing the C_L-C_D curves. This is done with the help of the equations presented in Section 2.5. The test data is presented in Table 3.4 and the derived parameters are presented in Table 3.5.

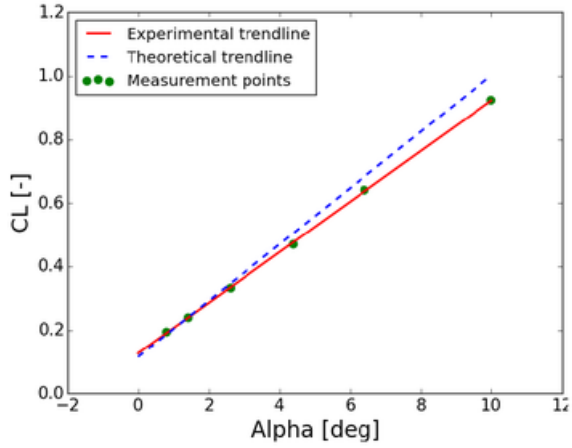
$$C_L = C_{L_\alpha}(\alpha - \alpha_0) \quad ; \quad C_D = C_{D_0} + \frac{C_L^2}{\pi A e} \tag{3.2}$$

Table 3.4: Measured data during first stationary measurement

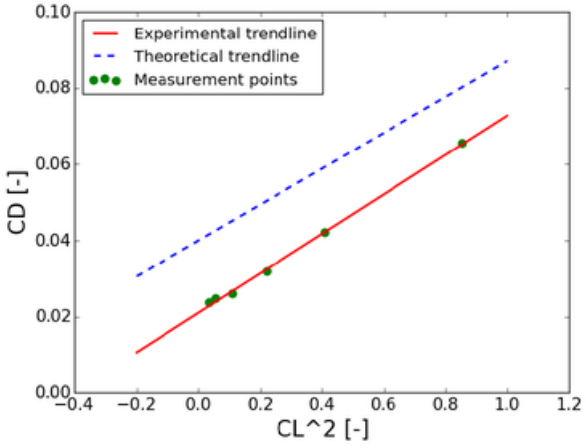
	Data points					
Parameter	**1**	**2**	**3**	**4**	**5**	**6**
h_p [m]	1527.048	1527.048	1527.048	1524	1527.048	1539.24
IAS [m/s]	129.125	115.750	98.258	82.311	70.478	58.646
α [°]	0.8	1.4	2.6	4.4	6.4	10.0
FFl [kg/s]	0.095	0.081	0.065	0.056	0.053	0.052
FFr [kg/s]	0.103	0.089	0.071	0.063	0.060	0.061
F_{used} [kg]	143.789	158.303	170.550	180.076	189.601	199.580
TAT [K]	284	283.2	281.5	279.8	279	277.2

Table 3.5: Calculated parameters from first measurement series

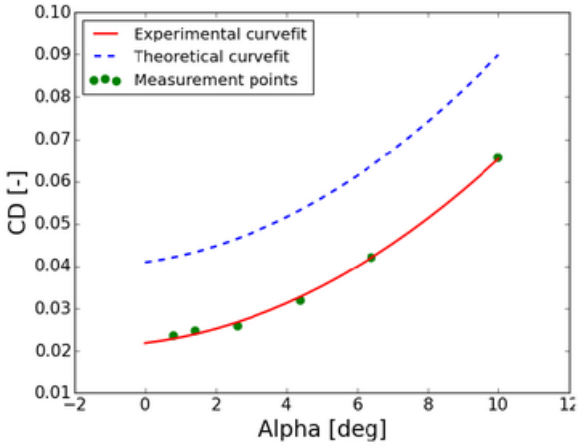
	Data points					
Parameter	**1**	**2**	**3**	**4**	**5**	**6**
p [Pa]	84275.563	84275.563	84275.563	84307.109	84275.563	84149.472
M	0.415	0.372	0.316	0.265	0.227	0.189
T [K]	274.559	275.576	275.989	275.930	276.157	275.234
ρ [kg/m^3]	1.069	1.065	1.063	1.064	1.063	1.065
a [m/s]	332.170	332.784	333.034	332.998	333.135	332.578
V_t [m/s]	137.727	123.770	105.226	88.174	75.573	62.847
V_e [m/s]	128.678	115.425	98.057	82.192	70.403	58.602
R_e	1.758e+7	1.569e+7	1.330e+7	1.116e+7	9.54e+6	7.97e+6
$W_{inst.}$ [kg]	6000.080	5985.565	5973.318	5963.793	5954.267	5944.288
C_L	0.193	0.239	0.331	0.471	0.641	0.923
C_D	0.023	0.024	0.025	0.031	0.042	0.065



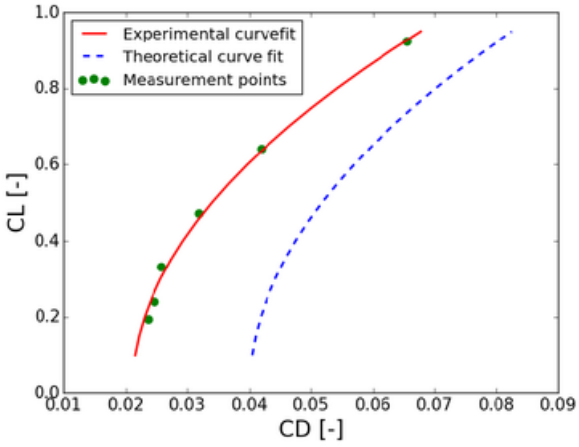
(a) C_L vs. α



(b) C_D vs. C_L^2



(c) C_D vs. α



(d) C_L vs. C_D

Figure 3.1: Plots of aerodynamic characteristics

Now that all the data is gathered, it is possible to make the necessary plots. The data points are curve fitted using polynomial regression. The coefficients are then matched with Equation 3.2 giving lift slope (C_{L_α}), zero lift-drag coefficient (C_{D_0}) and Oswald factor (e). Figure 3.1 consists of C_D - C_L^2, C_L - α C_L - C_D and C_D - α graphs.

- In Figure 3.1a, the lift coefficient is plotted against the angle of attack. It can be observed that there is a linear trend and from literature, it is known that the lift curve is mostly linear upto the point of stall. The aircraft did not reach the stalling point at the time of stationary measurements and hence, the non-linear part is not seen. The theoretical line is plotted using the data provided in the assignment. However, the zero-lift angle of attack was obtained from the airfoil series, since it is the same for the aircraft. It can be noticed that the lift gradient of the experimental line is lower than that of the theoretical line. This deviation is perhaps due to the 3-Dimensional effects and small angle approximations.

- In Figure 3.1b, the drag coefficient is plotted against the squared lift coefficient and there exists a linear correlation between the two. This is an expected trend and in agreement with theory as can be confirmed from the lift-drag polar equation (see Equation 3.2). Figure 3.1c is a plot of lift coefficient as a function of drag coefficient and again, as expected, a quadratic relation. Also, in Figure 3.1d, the drag coefficient is plotted as a function of angle of attack and as expected, there is a parabolic trend. A remarkable difference with theory is observed in these three plots. The minimum value of the drag coefficient, i.e., the zero-lift drag coefficient occurs when the lift coefficient goes to zero. This value for experimental is much higher than that of the theoretical. A lower values suggests that in reality, there is alot more induced drag produced. For the same reason, the Oswald factor comes out to be lower when compared to theory. Table 3.7 provides a detailed comparison to theory and quantifies the introduced deviation.

The simplified assumptions made at the time of deriving the equilibrium equations of motion have a large effect in the results. In reality, the flight is never perfectly steady, symmetric or horizontal and in addition, the thrust vector acts at an angle w.r.t the nose. Including these effects in the analysis reflects the lift and drag values due to the extra contributions to the total horizontal and vertical forces.

Table 3.6: Measured data of second measurement series

	Data points									
Parameter	**1**	**2**	**3**	**4**	**5**	**6**	**7**		**8**	**9**
h_p [m]	2145.79	2225.04	2325.62	2432.04	2185.41	2033.01	1889.76	**C**	2048.25	2078.73
IAS [m/s]	87.45	82.31	77.16	72.53	94.14	97.74	101.85	**G**	87.45	87.45
α [°]	3.6	4.4	5.2	5.9	2.9	2.5	2.2		3.6	3.6
δ_e[°]	-0.2	-0.5	-0.9	-1.3	0.2	0.4	0.6	**S**	-0.1	-0.6
$\delta_{t,e}$[°]	3.3	3.3	3.3	3.3	3.3	3.3	3.3	**h**	3.3	3.3
F_e [N]	0	-22	-34	-42	32	52	70	**i**	0	-27
FFl [kg/s]	0.055	0.055	0.054	0.054	0.056	0.057	0.058	**f**	0.056	0.056
FFr [kg/s]	0.061	0.061	0.060	0.060	0.062	0.063	0.064	**t**	0.062	0.062
F_{used} [kg]	243.57	250.38	258.54	264.44	277.59	293.47	300.73		311.61	324.31
TAT [K]	277.5	276.5	275	274.2	277.8	279	280.5		277.5	277.8

3.2.2. Second Series

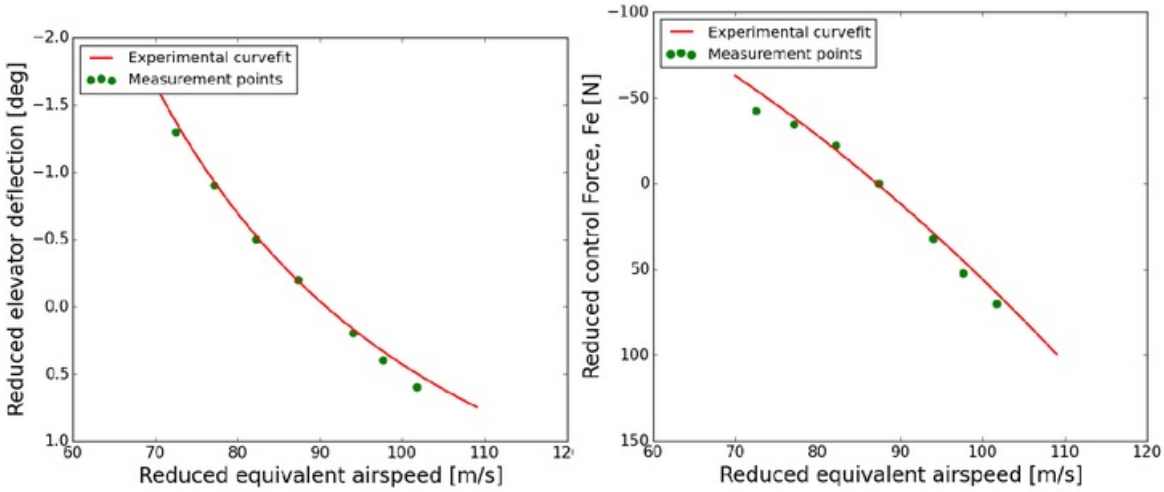
The data acquired from this measurement set is used to derive the longitudinal stability derivatives and to study the effect of airspeed on elevator deflection and on control force. The measured values during the test are presented in Table 3.6. The procedure followed to determine the elevator effectiveness and longitudinal stability is described in Section 2.5, more specifically refer Subsection 2.5.3. The elevator trim and control force curves follow from there and are plotted in Figure 3.2

3.3. Eigenmotion Measurements

After the stationary measurements were completed, the next part of the flight test was the investigation into the characteristic motions of the aircraft. The Flight Test Instrumentation System (FTIS) recorded the control

Table 3.7: Flight test synopsis

Ouputs	Experimental value	Theoretical value	Percentage deviation
First series			
e	0.73	0.8	8.75%
α_0 [°]	-1.59	-1.32	20.4%
C_{D_0}	0.02	0.04	50%
C_{L_α} [/rad]	4.56	5.08	10.2%
Second series			
C_{m_δ} [/rad]	-1.42	-1.16	22.4%
C_{m_α} [/rad]	-0.70	-0.56	26.25%



(a)

(b) Graphs of reduced elevator deflection and control Force, with respect to the reduced equivalent airspeed

Figure 3.2

inputs made, namely control surface deflections, together with the aircraft response outputs, such as true airspeed, but also environmental data, such as pressure altitude. This data provided the means for determining relevant parameters in the design of the analytical and numerical models, as well as the basis for validation of the computational model. An example of some of the obtained data arrays can be seen as plotted in Figure 3.3.

3.4. Missing Data

In case the experiment had an error or was not done properly, a basic alternative was established. Assuming that some of the data is missing, this should not be a problem. Referring to the flowchart, the inputs are divided into three groups, the aircraft parameters, atmospheric constants and flight data. In case, the aircraft parameters are missing, the value could be taken from reference aircraft or the data provided by the manufacturer. This wouldn't have a very drastic effect on the final result. Moreover, for the atmospheric constants, the same approach can be taken place too. If anyone of the atmospheric constants or doesn't make sense, literature values can be refereed to double check the values. Changing this would not effect the final result very much, as the atmospheric constants gotten from the flight test are similar to the literature value. However, in case of missing data or of low quality of flight data, the data could be taken from other test simulation with similar scenarios. As the experiment was done several times, the data could be taken from other ones with the most similar conditions. But, this would effect the final data as the conditions the flight are taking

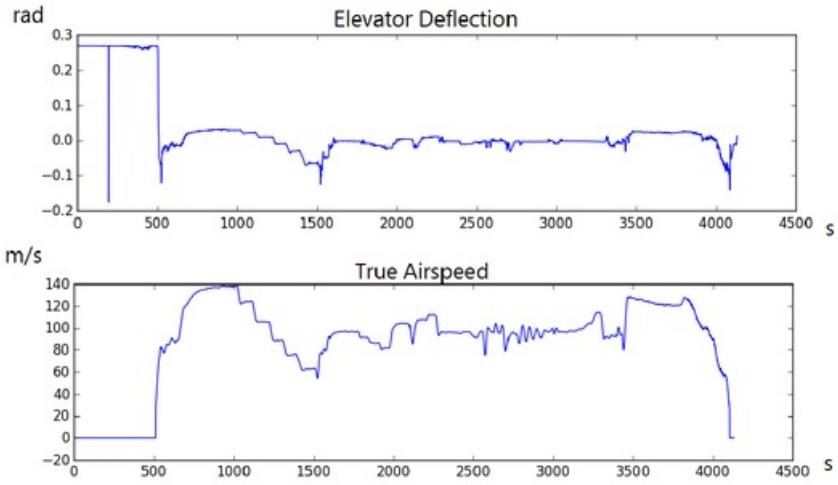


Figure 3.3: Example data sets: elevator deflection and true airspeed

place might be different, thus effecting the final results. For instance, the fuel weight of different flights are different, and so, this effects the centre of gravity.

Chapter 4

Analytical Model

The process of building the analytical model began with the linearised equations of motion, as derived using the assumptions of Section 2.4 [5]. In order to obtain an analytical solution, the linearised equations of motion were simplified further. The models for each eigenmotion of interest are presented in Section 4.1 for symmetric ones and in Section 4.2 for asymmetric ones. Certain coefficients, such as C_{m_α}, were determined from the experimental data. For the majority of coefficients, however, typical values were sourced initially [15]; the most important coefficients were identified for tweaking the numerical model later on. The solutions, Section 4.3, were found by using the same control inputs used in the flight test; these solutions provided the basis for system testing the computation model during the verification process. Further assumptions were made in order to simplify the equations for the specific eigenmotions under consideration; their effects are discussed in Section 4.4.

4.1. Symmetric Equations of Motion

The matrix form of the symmetric equation of motions is shown in Equation 4.1. The equations for the short period and phugoid eigenmotions were derived based on relevant assumptions pertaining to each one.

$$\begin{bmatrix} C_{X_u} - 2\mu_c D_c & C_{X_\alpha} & C_{Z_0} & 0 \\ C_{Z_u} & C_{Z_\alpha} + (C_{Z_{\dot{\alpha}}} - 2\mu_c) D_c & -C_{X_0} & C_{Z_q} + 2\mu_c \\ 0 & 0 & -D_c & 1 \\ C_{m_u} & C_{m_\alpha} + C_{m_{\dot{\alpha}}} D_c & 0 & C_{m_q} - 2\mu_c K_Y^2 D_c \end{bmatrix} \begin{bmatrix} \hat{u} \\ \alpha \\ \theta \\ \frac{q\bar{c}}{V} \end{bmatrix} = 0 \tag{4.1}$$

4.1.1. Short Period

Short period motion is caused by a sudden deflection in the elevator that causes a change in pitch rate, this motion is heavily damped and typically only lasts a couple seconds. The short period is a symmetric flight motion thus the symmetric equation of motion, Equation 4.1, is applicable. This set of equations can simplified by assuming the the velocity is constant, this is valid as the motion occurs over such a short time period

that the velocity does not have time to change. In reality the velocity will vary slightly but this is assumed to be negligible for this analysis. The implication of this is that $\hat{u} = 0$; thus the first column of Equation 4.1 becomes equal to zero. This also leads to a force equilibrium in the X_B direction; thus, that equation can be omitted. Additionally, it is assumed that the aircraft is initially in steady flight, meaning $\gamma_0 = 0$ and $C_{X_0} = 0$. These allow omission of the pitch angle, θ, as well. The matrix for the short period eigenmotion becomes Equation 4.2.

$$\begin{bmatrix} C_{Z_\alpha} + (C_{Z_{\dot{\alpha}}} - 2\mu_e)D_C & C_{Z_q} + 2\mu_c \\ C_{m_\alpha} + C_{m_{\dot{\alpha}}} D_c & C_{m_q} - 2\mu_c K_y^2 D_c \end{bmatrix} \begin{bmatrix} \alpha \\ \frac{q\bar{c}}{V} \end{bmatrix} = 0 \tag{4.2}$$

The coefficients which will impact this eigenmotion the most are C_{z_α} and C_{m_q}. The reason that these are the main coefficients is because μ_c is significantly larger than C_{z_q} and $C_{z_{\dot{\alpha}}}$. Taking into account that C_{m_α} and μ_c are not tweakable, since they depend on the performance curve and flight parameters, only the aforementioned coefficients are useful to alter in order to achieve a better model.

4.1.2. Phugoid

The phugoid eigenmotion is a long period oscillatory movement that is symmetric; thus Equation 4.1 is applicable. The system of equations can be simplified by assuming that due to the long period of the motion, the rate of change of the angle of attack is negligible: $\dot{\alpha} = 0$. This means that terms with $D_C\alpha$ are equal to zero. Furthermore, the pitch rate of change is assumed to be very slow; hence, the rate of change of the pitch rate,$\dot{q}$, is neglected. Similar to the short period motion, the initial steady flight condition is assumed, meaning $C_{X_0} = 0$. Also, C_{Z_q} is neglected since, relative to the mass parameter $2\mu_c$, it is very small. This combines to simplify the initial equations of motion to the equations of motion presented in Equation 4.3.

$$\begin{bmatrix} C_{X_u} - 2\mu_c D_c & C_{X_\alpha} & C_{Z_0} & 0 \\ C_{Z_u} & C_{Z_\alpha} & 0 & 2\mu_c \\ 0 & 0 & -D_c & 1 \\ C_{m_u} & C_{m_\alpha} & 0 & C_{m_q} \end{bmatrix} \begin{bmatrix} \hat{u} \\ \alpha \\ \theta \\ \frac{q\bar{c}}{V} \end{bmatrix} = 0 \tag{4.3}$$

From the above matrix it is more difficult to identify the key coefficients. C_{x_α}, C_{z_u}, and C_{m_u} are negligibly small and are therefore initially not considered important to tweak. However, the two parameters, C_{z_α} and C_{m_q}, have already been established as important parameters for tweaking the short period response and are present in this matrix as well. Tweaking these parameters would affect both eigenmotions, so the previously mentioned coefficients may also need to be adjusted.

4.2. Asymmetric Equations of Motion

The matrix form of the asymmetric equation of motions is shown in Equation 4.4. Equations for the aperiodic roll, Dutch roll, and aperiodic spiral were derived based on relevant assumptions pertaining to each one.

$$\begin{bmatrix} C_{Y_\beta} + (C_{Y_{\dot{\beta}}} - 2\mu_b)D_b & C_L & C_{Y_p} & C_{Y_r} - 4\mu_b \\ 0 & -\frac{1}{2}D_b & 1 & 0 \\ C_{l_\beta} & 0 & C_{l_p} - 4\mu_b K_X^2 D_b & C_{l_r} + 4\mu_b K_{XZ} D_b \\ C_{n_\beta} + C_{n_{\dot{\beta}}} D_b & 0 & C_{n_p} + 4\mu_b K_{XZ} D_b & C_{n_r} - 4\mu_b K_Z^2 D_b \end{bmatrix} \begin{bmatrix} \beta \\ \phi \\ \frac{pb}{2V} \\ \frac{rb}{2V} \end{bmatrix} = 0 \tag{4.4}$$

4.2.1. Aperiodic Roll

The aperiodic roll eigenmotion is an asymmetric motion as the aircraft rolls about the longitudinal axis; thus, the asymmetric equations of motion, Equation 4.4, are applicable in this case. For this motion, it is assumed that the aircraft only rolls; other motions are ignored. This is possible as, typically, the roll damps out before other eigenmotions start to occur. This means that both the equations used to find the sideslip angle, β, and the non-dimensional yaw-rate, $\frac{rb}{2V}$, disappear. The third row in the original matrix for the asymmetric equations of motion is independent of the roll angle, ϕ; thus, the relation $-\frac{1}{2}D_b\phi + \frac{pb}{2V} = 0$ is irrelevant. This leaves one equation to describe the aperiodic roll, Equation 4.5.

$$(C_{l_p} - 4\mu_b K_X^2 D_b)\frac{pb}{2V} = 0 \tag{4.5}$$

For this eigenmotion, only one coefficient can be tweaked, namely C_{l_p}.

4.2.2. Dutch Roll

Dutch roll describes the motion an aircraft undergoes when it yaws to the left and right in combination with a rolling motion. However, for most aircraft the Dutch roll motion can be described sufficiently by the yawing components; thus, the rolling components, ϕ and $\frac{pb}{2V}$, are set to zero. In reality, there is an effect due to roll, which will likely to cause small deviations in the results. This means that the second and third column in the original equations of motion (Equation 4.4) disappear. Furthermore, the third row in the same equation can be ignored as the rolling moment is no longer required. Again, C_{Y_β}, C_{n_β}, and C_{Y_r} are insignificant compared to the mass parameter $4\mu_b$ and are thus ignored as well. This leaves Equation 4.6, which describes the Dutch roll motion of the aircraft.

$$\begin{bmatrix} C_{Y_\beta} - 2\mu_b D_b & -4\mu_b \\ C_{n_\beta} & C_{n_r} - 4\mu b K_Z^2 D_b \end{bmatrix} \begin{bmatrix} \beta \\ \frac{rb}{2V} \end{bmatrix} = 0 \tag{4.6}$$

For the dutch roll, the coefficients that can be tweaked are C_{n_β}, C_{n_r}, and C_{Y_β}. As can be seen in Subsection 4.2.3, C_{n_β} and C_{n_r} affect both the aperiodic spiral and the Dutch roll.

4.2.3. Aperiodic Spiral

The aperiodic spiral is the eigenmotion for which an aircraft simultaneously rolls, yaws, and slips in a downward spiral trajectory. As the aperiodic spiral motion is a relatively slow motion, it can be assumed that all accelerations, except for the rate of change of yaw angle, are equal to zero. This means that $D_b\beta = D_b\frac{pb}{2V} = D_b\frac{rb}{2V} = 0$.

$$\begin{bmatrix} C_{Y_\beta} & C_L & 0 & -4\mu_b \\ 0 & -\frac{1}{2}D_b & 1 & 0 \\ C_{l_\beta} & 0 & C_{l_p} & C_{l_r} \\ C_{n_\beta} & 0 & C_{n_p} & C_{n_r} \end{bmatrix} \begin{bmatrix} \beta \\ \psi \\ \frac{pb}{2V} \\ \frac{rb}{2V} \end{bmatrix} = 0 \tag{4.7}$$

This matrix results in Equation 4.8 to find the eigenvalue. For stability, the numerator of this equation must be negative to ensure the eigenvalue is also negative (the denominator is always positive); therefore, the key coefficients are C_{l_β}, C_{n_r}, C_{n_β} and C_{l_r}. Also, because the first term in the brackets of the numerator consists of two negative values and the second term consists of two positive values, the magnitude of the second term must be larger than the first.

$$\lambda = \frac{2 * C_L(C_{l_\beta}C_{n_r} - C_{n_\beta}C_{l_r})}{C_{l_p}(C_{Y_\beta}C_{n_r} + 4\mu_b C_{n_\beta}) - C_{n_p}(C_{Y_\beta}C_{l_r} + 4\mu_b C_{l_\beta})} \tag{4.8}$$

4.3. Analytical Results

From the simplified equations of motions found in Section 4.1 and Section 4.2, different parameters can be calculated; these provide information about the stability and response of an aircraft to a specific eigenmotion. An eigenvalue with a large magnitude of the real part will mean that it is more heavily damped. In Table 4.1, the eigenvalues, time period, damping ratio, time to half amplitude, natural frequency, and time constant are presented.

$$\lambda_c = \varepsilon_c \pm j\eta_c \tag{4.9}$$

The parameters previously mentioned can be calculated using Equation 4.10 and Equation 4.13, where the real and imaginary parts of each eigenvalue are denoted as shown in Equation 4.9.

$$P = \frac{2\pi}{\eta_c} \tag{4.10}$$

$$T_{1/2} = \frac{ln\frac{1}{2}}{\varepsilon_c} \tag{4.11}$$

$$\omega_n = \sqrt{{\varepsilon_c}^2 + {\eta_c}^2} \tag{4.12}$$

$$\zeta = \frac{-\varepsilon_c}{\sqrt{{\varepsilon_c}^2 + {\eta_c}^2}} \tag{4.13}$$

The two symmetric motions, short period and phugoid, both have complex eigenvalues. The real part of the eigenvalue determines the stability and was in both cases negative, indicating that both eigenmotions are stable. As expected, the short period motion was heavily damped, indicated by a damping ratio of 0.534 and a period that lasted only 2.555 seconds. The results for the phugoid eigenmotion also concur with the expected output: the phugoid motion is slow and only lightly damped, showing a period of 46.987 seconds and a damping ratio of only 0.046. The asymmetric motions also showed the expected eigenvalues, both aperiodic roll and dutch roll have stable eigenvalues, as the real parts were negative. The aperiodic spiral eigenvalue indicated an unstable motion. Again, this corresponds to the expectation since aperiodic spiral

Table 4.1: Numerical eigenvalues and characteristics

Eigenmotion	Eigenvalue	Period [s]	ζ [-]	$T_{1/2}$[s]	ω_n [rad/s]	τ [-]
Symmetric						
Short Period	-1.552 ± 2.460j	2.555	0.534	0.446	2.460	-
Phugoid	-0.006 ± 0.134j	46.987	0.046	112.620	0.134	-
Asymmetric						
Aperiodic Roll	-5.049	-	1	0.137	5.049	0.032
Dutch Roll	-0.449 ± 2.410j	2.607	0.183	1.544	2.410	-
Aperiodic Spiral	0.0105	-	-1	-66.190	0.0105	-15.592

18

19 does not damp out over time; instead, it spirals over time. For both the aperiodic manoeuvres, the time constant, τ, was calculated since a time period cannot be determined.

20

4.4. Effects of Assumptions

Whilst deriving the simpler analytical equations of motion, additional assumptions were taken for each eigenmotion beyond the general assumptions explained in Section 2.4. These assumptions differed for each eigenmotion as each motion is inherently different and must be looked at individually. Although these assumptions were taken to be valid throughout the analytical model, they will in fact cause slight discrepancies in the outputs compared to using the complete equations of motions. It is, however, expected that the assumptions taken do not cause an error greater than 10% when compared to using the complete equations of motion. For example, in the short period motion, the change in velocity is assumed zero due to the short duration of the motion. In reality, the velocity will likely vary slightly and thus cause a small discrepancy in the final result. Similar reasoning is used for the other eigenmotions and thus the effects are considered small enough for the analytical method to be utilised.

Chapter 5

Numerical Model

21

This chapter outlines the numerical approach used to create the model of the aircraft's motion. This is the state space form of the two types of flight, namely symmetric flight and asymmetric flight. The goal of the numerical model was to create a model which accurately describes the dynamic behaviour of an aircraft.

In order to make the required eigenvalues for the symmetric and asymmetric flight, Python was used. In case of the numerical model, there is no additional simplifying assumptions taken as was done in the analytical approach; the general assumptions from Section 2.4 are the ones made for the numerical approach.

5.1. State-Space Form of Symmetric Equations

The linearised equations of motion for symmetric motion is given in Equation 5.1. However, in order to calculate the necessary outputs for the required simulation, a state-space model must be created for the symmetric flight cases. The state-space model requires the construction of four matrices and is set up with Equation 5.2, where A is the state matrix; B is the input matrix; C is the output matrix; and D is the feedthrough matrix. Moreover, $\bar{x}$ is the state vector, $\bar{u}$ is the input vector and $\bar{y}$ is the output vector [5].

$$\begin{bmatrix} C_{X_u} - 2\mu_c D_c & C_{X_\alpha} & C_{Z_0} & C_{X_q} \\ C_{Z_u} & C_{Z_\alpha} + (C_{Z_{\dot{\alpha}}} - 2\mu_c) D_c & -C_{X_0} & C_{Z_q} + 2\mu_c \\ 0 & 0 & -D_c & 1 \\ C_{m_u} & C_{m_\alpha} + C_{m_{\dot{\alpha}}} D_c & 0 & C_{m_q} - 2\mu_c K_Y^2 D_c \end{bmatrix} \begin{bmatrix} \hat{u} \\ \alpha \\ \theta \\ \frac{q\bar{c}}{V} \end{bmatrix} = \begin{bmatrix} C_{X_{\delta_e}} \\ C_{Z_{\delta_e}} \\ 0 \\ C_{m_{\delta_e}} \end{bmatrix} \cdot \delta_e \tag{5.1}$$

$$\dot{\bar{x}} = A\bar{x} + B\bar{u} \quad \bar{y} = C\bar{x} + D\bar{u} \tag{5.2}$$

From the equations of motion, Equation 5.3 can be created, from which the matrices A and B can be determined with Equation 5.4.

$$C_1\dot{\bar{x}} + C_2\bar{x} + C_3\bar{u} = \bar{0} \tag{5.3}$$

$$A = -C_1^{-1}\cdot C_2 \quad B = -C_1^{-1}\cdot C_3 \tag{5.4}$$

The matrices, C_1, C_2 and C_3, can be derived from Equation 5.1 by moving all terms without the differential operator, D_c, to the right hand side; D_c is then replaced by $\frac{\bar{c}}{V}\frac{d}{dt}$ [5]. This manipulation allows the equations to be written in the form of Equation 5.3. For symmetric flight, C_1, C_2 and C_3 were hence obtained as Equation 5.5, Equation 5.6, and Equation 5.7.

$$C_1 = \begin{bmatrix} -2\mu_c D_c & 0 & 0 & 0 \\ 0 & (C_{Z_{\dot{\alpha}}} - 2\mu_c)D_c & 0 & 0 \\ 0 & 0 & -D_c & 0 \\ 0 & C_{m_{\dot{\alpha}}} D_c & 0 & -2\mu_c K_Y^2 D_c \end{bmatrix} \tag{5.5}$$

$$C_2 = \begin{bmatrix} C_{X_u} & C_{X_\alpha} & C_{Z_0} & C_{X_q} \\ C_{Z_u} & C_{Z_\alpha} & -C_{X_0} & C_{Z_q} + 2\mu_c \\ 0 & 0 & 0 & 1 \\ C_{m_u} & C_{m_\alpha} & 0 & C_{m_q} \end{bmatrix} \tag{5.6}$$

$$C_3 = \begin{bmatrix} C_{X_{\delta_e}} \\ C_{Z_{\delta_e}} \\ 0 \\ C_{m_{\delta_e}} \end{bmatrix} \tag{5.7}$$

By obtaining the values of C_1, C_2 and C_3, the matrices A and B are shown in Equation 5.8 and Equation 5.9. Note that D_C is the differential operator and can be substituted by $\frac{\bar{c}}{V}\frac{d}{dt}$.

$$A = \begin{bmatrix} \frac{V}{\bar{c}}\cdot\frac{C_{X_u}}{2\mu_c} & \frac{V}{\bar{c}}\cdot\frac{C_{X_\alpha}}{2\mu_c} & \frac{V}{\bar{c}}\cdot\frac{C_{Z_0}}{2\mu_c} & \frac{V}{\bar{c}}\cdot\frac{C_{X_q}}{2\mu_c} \\ \frac{V}{\bar{c}}\cdot\frac{C_{Z_u}}{2\mu_c - C_{Z_{\dot{\alpha}}}} & \frac{V}{\bar{c}}\cdot\frac{C_{Z_\alpha}}{2\mu_c - C_{Z_{\dot{\alpha}}}} & -\frac{V}{\bar{c}}\cdot\frac{C_{X_0}}{2\mu_c - C_{Z_{\dot{\alpha}}}} & \frac{V}{\bar{c}}\cdot\frac{C_{Z_q}+2\mu_c}{2\mu_c - C_{Z_{\dot{\alpha}}}} \\ 0 & 0 & 0 & \frac{V}{\bar{c}} \\ \frac{V}{\bar{c}}\cdot\frac{C_{m_u} + C_{Z_u}\cdot\frac{C_{m_{\dot{\alpha}}}}{2\mu_c - C_{Z_{\dot{\alpha}}}}}{2\mu_c K_Y{}^2} & \frac{V}{\bar{c}}\cdot\frac{C_{m_\alpha} + C_{Z_\alpha}\cdot\frac{C_{m_{\dot{\alpha}}}}{2\mu_c - C_{Z_{\dot{\alpha}}}}}{2\mu_c K_Y{}^2} & -\frac{V}{\bar{c}}\cdot\frac{C_{X_0}\cdot\frac{C_{m_{\dot{\alpha}}}}{2\mu_c - C_{Z_{\dot{\alpha}}}}}{2\mu_c K_Y{}^2} & \frac{V}{\bar{c}}\cdot\frac{C_{m_q} + C_{m_{\dot{\alpha}}}\cdot\frac{2\mu_c + C_{Z_q}}{2\mu_c - C_{Z_{\dot{\alpha}}}}}{2\mu_c K_Y{}^2} \end{bmatrix} \tag{5.8}$$

$$B = \begin{bmatrix} \frac{V}{\bar{c}}\cdot\frac{C_{X_{\delta_e}}}{2\mu_c} \\ \frac{V}{\bar{c}}\cdot\frac{C_{Z_{\delta_e}}}{2\mu_c - C_{Z_{\dot{\alpha}}}} \\ 0 \\ \frac{V}{\bar{c}}\cdot\frac{C_{Z_{\delta_e}}\cdot\frac{C_{m_{\dot{\alpha}}}}{2\mu_c - C_{Z_{\dot{\alpha}}}}}{2\mu_c K_Y{}^2} \end{bmatrix} \tag{5.9}$$

As C and D are dependent on the required output vector, and the desired outputs are equal to the state variables, these matrices were taken as, respectively, a unit matrix and a null matrix, as shown in Equation 5.10 and Equation 5.11. This is a very simple way of obtaining the desired output.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.10}$$

$$D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.11}$$

5.2. State-Space Form of Asymmetric Equations

The same approach can be taken for the asymmetric motion as was taken for the symmetric motion [5]. The linearised equations of motion for asymmetric motion are given in Equation 5.12.

$$\begin{bmatrix} C_{Y_\beta} + (C_{Y_{\dot{\beta}}} - 2\mu_b)D_b & C_L & C_{Y_p} & C_{Y_r} - 4\mu_b \\ 0 & -\frac{1}{2}D_b & 1 & 0 \\ C_{l_\beta} & 0 & C_{l_p} - 4\mu_b K_X^2 D_b & C_{l_r} + 4\mu_b K_{XZ} D_b \\ C_{n_\beta} + C_{n_{\dot{\beta}}} D_b & 0 & C_{n_p} + 4\mu_b K_{XZ} D_b & C_{n_r} - 4\mu_b K_Z^2 D_b \end{bmatrix} \begin{bmatrix} \beta \\ \phi \\ \frac{pb}{2V} \\ \frac{rb}{2V} \end{bmatrix} = \begin{bmatrix} C_{Y_{\delta a}} \\ 0 \\ C_{l_{\delta a}} \\ C_{n_{\delta a}} \end{bmatrix} \cdot \delta_a + \begin{bmatrix} C_{Y_{\delta r}} \\ 0 \\ C_{l_{\delta r}} \\ C_{n_{\delta r}} \end{bmatrix} \cdot \delta_r \tag{5.12}$$

Using Equation 5.3, for asymmetric flight, C_1; C_2; and C_3 are Equations 5.13; 5.14; and 5.15, respectively.

$$C_1 = \begin{bmatrix} (C_{Y_{\dot{\beta}}} - 2\mu_b)D_b & 0 & 0 & 0 \\ 0 & -\frac{1}{2}D_b & 0 & 0 \\ 0 & 0 & -4\mu_b K_X^2 D_b & 4\mu_b K_{XZ} D_b \\ C_{n_{\dot{\beta}}} D_b & 0 & 4\mu_b K_{XZ} D_b & -4\mu_b K_Z^2 D_b \end{bmatrix} \tag{5.13}$$

$$C_2 = \begin{bmatrix} C_{Y_\beta} & C_L & C_{Y_p} & C_{Y_r} - 4\mu_b \\ 0 & 0 & 1 & 0 \\ C_{l_\beta} & 0 & C_{l_p} & C_{l_r} \\ C_{n_\beta} & 0 & C_{n_p} & C_{n_r} \end{bmatrix} \quad (5.14) \qquad C_3 = \begin{bmatrix} C_{Y_{\delta_a}} & C_{Y_{\delta_r}} \\ 0 & 0 \\ C_{l_{\delta_a}} & C_{l_{\delta_r}} \\ C_{n_{\delta_a}} & C_{n_{\delta_r}} \end{bmatrix} \quad (5.15)$$

Using Equation 5.4, matrices A and B were determined for the asymmetric equations in Equation 5.16 and Equation 5.17. Note that D_b is the differential operator and therefore can be substituted by $\frac{b}{V}\frac{d}{dt}$. For matrices C and D, similar reasoning is used as for Equation 5.10 and Equation 5.11, since in this case the desired outputs are equal to the state vector.

$$A = \begin{bmatrix} \frac{V}{b}\cdot\frac{C_{Y_\beta}}{2\mu_b} & \frac{V}{b}\cdot\frac{C_L}{2\mu_b} & \frac{V}{b}\cdot\frac{C_{Y_p}}{2\mu_b} & \frac{V}{b}\cdot\frac{C_{Y_r}-4\mu_b}{2\mu_b} \\ 0 & 0 & 2\frac{V}{b} & 0 \\ \frac{V}{b}\cdot\frac{C_{l_\beta}K_Z{}^2+C_{n_\beta}K_{XZ}}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & 0 & \frac{V}{b}\cdot\frac{C_{l_p}K_Z{}^2+C_{n_p}K_{XZ}}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & \frac{V}{b}\cdot\frac{C_{l_r}K_Z{}^2+C_{n_r}K_{XZ}}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} \\ \frac{V}{b}\cdot\frac{C_{l_\beta}K_{XZ}+C_{n_\beta}K_X{}^2}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & 0 & \frac{V}{b}\cdot\frac{C_{l_p}K_{XZ}+C_{n_p}K_X{}^2}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & \frac{V}{b}\cdot\frac{C_{l_r}K_{XZ}+C_{n_r}K_X{}^2}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} \end{bmatrix} \quad (5.16)$$

$$B = \begin{bmatrix} 0 & \frac{V}{b}\cdot\frac{C_{Y_{\delta_r}}}{2\mu_b} \\ 0 & 0 \\ \frac{V}{b}\cdot\frac{C_{l_{\delta_a}}K_Z{}^2+C_{n_{\delta_a}}K_{XZ}}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & \frac{V}{b}\cdot\frac{C_{l_{\delta_r}}K_Z{}^2+C_{n_{\delta_r}}K_{XZ}}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} \\ \frac{V}{b}\cdot\frac{C_{\delta_a}K_{XZ}+C_{n_{\delta_a}}K_X{}^2}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} & \frac{V}{b}\cdot\frac{C_{\delta_r}K_{XZ}+C_{n_{\delta_r}}K_X{}^2}{4\mu_b(K_X^2K_Z^2-K_{XZ}^2)} \end{bmatrix} \quad (5.17)$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.18) \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (5.19)$$

5.3. Eigenvalues

As matrix A is the state matrix, it represents the system state. Thus, with the symmetric and asymmetric A matrix, the eigenvalues can be split into real and imaginary parts, and are projected in the complex plane. The eigenvalues for the symmetric state space system are depicted in Figure 5.1 and the asymmetric ones in Figure 5.2, where the x-axis represents the value of the real part of the eigenvalue and the y-axis represents the imaginary part of the eigenvalue.

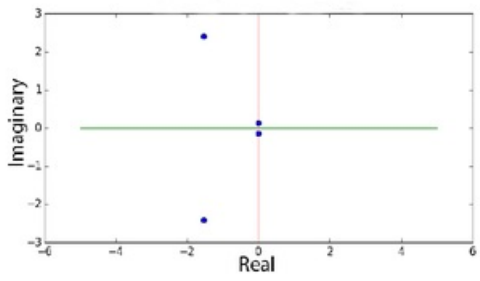


Figure 5.1: Graph of symmetric eigenvalues.

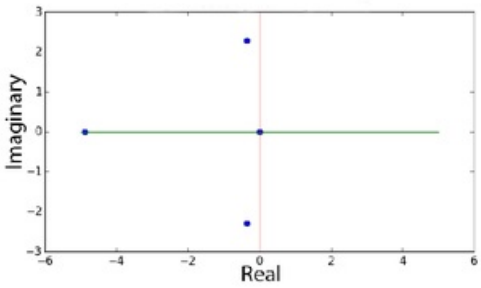


Figure 5.2: Graph of asymmetric eigenvalues.

22

From Figure 5.1, it was observed that there are two pairs of complex conjugate eigenvalues. One of these eigenvalues shows a much greater magnitude imaginary part; this means that the corresponding eigenmotion experiences a much shorter period compared to the other eigenmotion. Combined with the fact that it also has a greater magnitude in the real part (indicating more damping), it was apparent that this eigenvalue corresponds to the short period eigenmotion. The other complex conjugate pair was indicative of a very lightly damped, long time period motion due to the very small magnitude of the complex part of the eigenvalue. This pair of eigenvalues corresponds to the phugoid eigenmotion, which is characterised by low damping and a very long time period. For the asymmetric eigenvalues, shown in Figure 5.2, there is one pair

of complex conjugates and two real eigenvalues. First, the complex pair of eigenvalues were examined; due to the complex nature of the eigenvalue, it was known that the eigenmotion was periodic. Furthermore, it was known that the only asymmetric eigenmotion that was periodic was the Dutch roll; hence Dutch roll corresponds to the complex pair of eigenvalues. The two remaining eigenvalues are both real however; thus, it is known that they are both aperiodic. The key difference between the eigenvalues is that one is negative whilst the other is positive and thus unstable. It is known that the aperiodic spiral eigenmotion is an unstable motion and thus the positive eigenvalue corresponds to this motion whilst the negative real eigenvalue coincides with the aperiodic roll eigenmotion [5].

The eigenvalues and some key characteristics are shown in Table 5.1; the characteristics are obtained using the equations explained in Section 4.3.

Table 5.1: Numerical eigenvalues and their characteristics.

Eigenmotion	Eigenvalue	Period [s]	ζ [-]	$T_{\frac{1}{2}}$ [s]	ω_n [rad/s]	τ [-]
Symmetric						
Short Period	-1.552 ± 2.453j	2.561	0.535	0.447	2.453	-
Phugoid	-0.0043 ± 0.135j	46.522	0.032	161.684	0.135	-
Asymmetric						
Aperiodic Roll	-5.136	-	1	0.135	5.136	0.031
Dutch Roll	-0.393 ± 2.499j	2.515	0.156	1.762	2.499	-
Aperiodic Spiral	0.0103	-	-1	-67.167	0.0103	-15.822

Chapter 6

23

Computational Model

For producing simulation results from the numerical model, the model must be implemented into code. To this end, a numerical algorithm was devised, Section 6.1. A flow chart outlines the algorithm graphically, Section 6.2. The same inputs as those of the test flight (Section 3.3) were used to produce results; the simulation output from this is presented in Section 8.3.

6.1. Numerical Algorithm

In order to construct a numerical algorithm or simulator, the best solution was to split the code into different modules. Each module has a specific task so as to keep a clear overview and enable easy detection of errors. These modules are: flight data reading; simulating the dynamical system; and, finally, looping the simulation to obtain outputs.

6.1.1. Flight Data Reading

The first module of the simulator is very short; it is responsible for reading the flight test data and outputting it to the different modules. It divides the data into different sections such as pilot/system inputs; aircraft/system outputs; and environmental data. It prepares them for throughput to the next module.

6.1.2. Simulation

In order to start simulating the aircraft response, first, system inputs and environmental data are imported from the flight data reader module. Combining this data with aircraft properties obtained from Section 3.1 and Section 3.2, and properties provided in the assignment, a mathematical dynamic system can be created, using methods described in Section 5.1 and Section 5.2. This system represents the aircraft and simulates the aircraft's response to given pilot inputs, the outputs describe the aircraft's motion. From this system, eigenvalues can also be obtained, representing the eigenmodes, and with these something can already be said about the stability of the aircraft. This information is used for aircraft stability analysis later.

6.1.3. Loop

Now the flight data is ready available, and a simulation can be made, the flight data is scanned and the eigenmodes, as performed by the pilots, are extracted from the full data set for further investigation. The control surface deflections during these eigenmodes are forwarded as input to the simulation described in Subsection 6.1.2, and from the simulation aircraft responses are imported. This process can be repeated for all eigenmodes by performing a loop, and finally the simulated responses can be plotted for further investigation and validation; this is shown in Section 8.3.

24

6.1.4. Tweaking

When the numerical model is completed, and when validation with measured data is performed, some discrepancies between simulated eigenmodes and real eigenmodes can occur. These discrepancies can be explained by a combination of factors, which is further explained in Subsection 8.4.1. However, to make the simulator resemble the real aircraft more closely, some parameters can be tweaked. These parameters are the stability derivatives, and can be directly linked to the eigenmodes of the aircraft. When a stability derivative can be linked to a certain eigenmode and is determined to have a significant effect on the shape of this eigenmode, it is considered a key stability derivative and can be tweaked to obtain better matching simulated responses. These key stability derivatives are determined in Section 4.1 and Section 4.2.

6.2. Flowchart

The flowchart that describes the algorithmic implementation of the simulation is presented in Figure 6.1.

Chapter 7

Verification

It is essential to determine whether the simulation devised accurately represents the physical model of the jet that was laid out earlier in Chapter 2 [16]. Code verification, Section 7.1, and unit testing, Section 7.2, were performed continually throughout the project. The distinction between these two verification processes lies with the numerical model: code verification checks that the model has been implemented correctly, whilst unit testing checks that the model calculates the correct output for each code module [3]. Finally, in Section 7.3, a system test was performed whereby the results of the analytical model were compared to the results of the numerical simulation for the same inputs.

7.1. Code Verification

An ongoing process, code verification involves the continual debugging of code [3]. This was initially done by looking into any syntactical and runtime errors identified by the compile when compiling the code; these errors were isolated and fixed. Further methods were employed, such as static analysis and binary searching, whereby sections of code were commented out and partial units were run, in order to identify mistakes in the implementation more easily. Whilst two four team members were involved in actively coding the simulation, the other two team members were used to perform this verification on the code objectively. define degree (e.g. 10%) reference data (literature, by hand etc..) for simpler problems

25

7.2. Unit Testing

It is important to check the individual modules, or 'units', of the simulation. With full system testing, errors can cancel each other out and go unnoticed. Furthermore, the continual process of unit testing allowed for errors to be detected early specific test cases. The unit testing is also able to provide a sanity check while coding the program to minimise the number of errors as much as possible.

7.2.1. Centre of Gravity Calculations

To make sure the centre of gravity calculations were done correctly, the centre of gravity were checked by considering the weight of everyone in the plane as zero and checking that the value of the centre of gravity

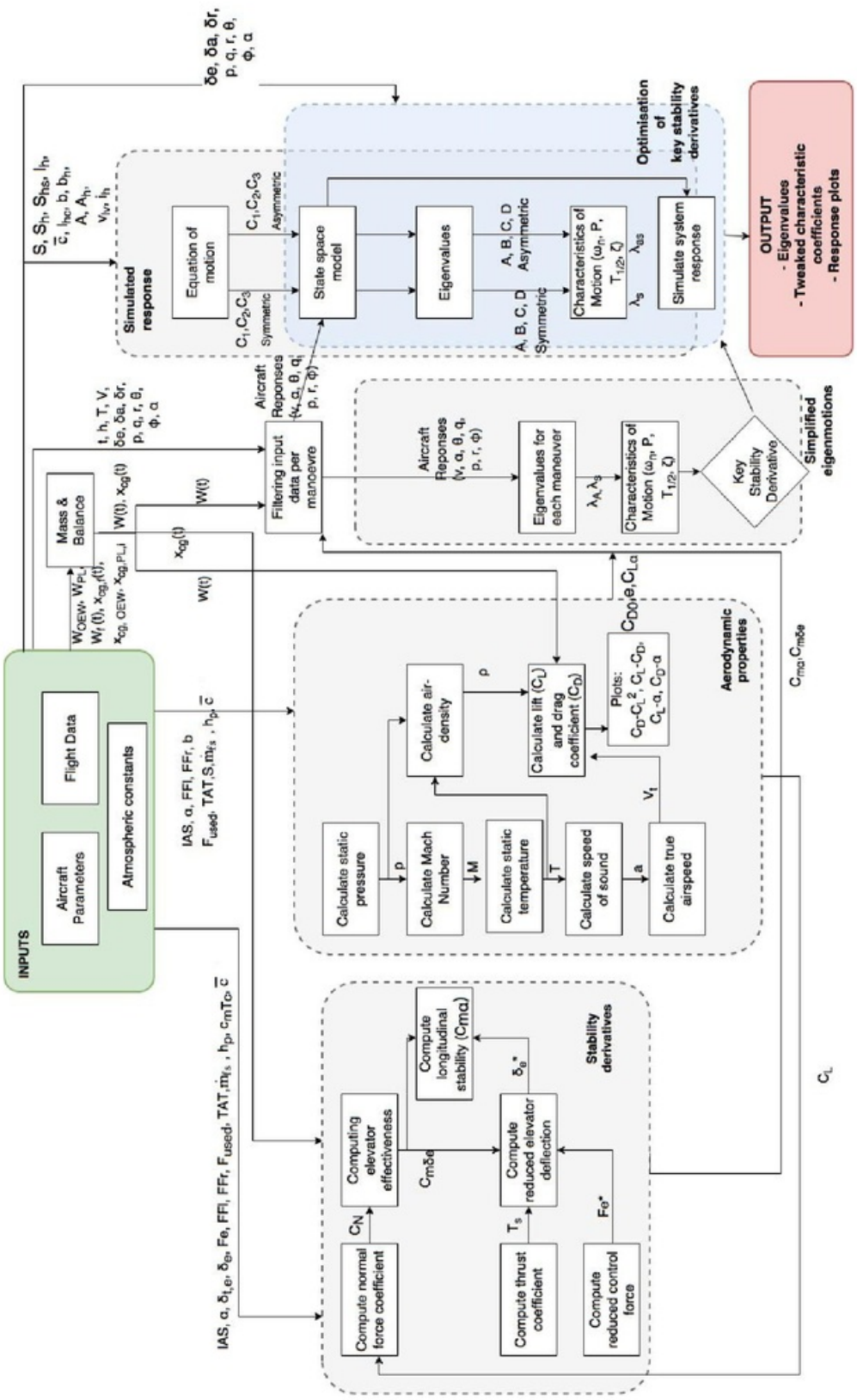


Figure 6.1: Code flowchart

in this case would coincide with the centre of gravity of the OEW, as provided by the manufacturer [15]. This would mean that the equation used to find the centre of gravity is correct. In addition, the weights of the individuals were changed to see if the centre of gravity would move in the expected direction.

7.2.2. Atmospheric Conditions

The atmospheric conditions that were calculated were checked with literature value in order to make sure that the values that were obtained by using the standard atmospheric model (ISA) made sense.

7.2.3. Aerodynamic Properties

To be able predict and compare the behavior of the aircraft aerodynamics, some important parameters like lift slope, zero-lift drag and Oswald factor were determined from the first measurement series. These values were checked with the theoretical results and also graphs have been constructed to verify the same. C_{L_α} is the slope of the lift-curve, C_{D_0} and e are obtained from the slope and y-intercept of the C_D- C_L^2 plot.

7.2.4. Stability Derivatives

Two stability derivatives were determined, the elevator effectiveness and the longitudinal stability, both from the second measurement series. These values were checked by manual calculations and also using graphical methods.

7.2.5. The State Space System

The state space system had to be created in order to get the required eigenvalues. In order to make the state space, first the C_1, C_2 and C_3 were created, after which, these coefficients were put in a Python program to give Matrix A and B, from which Matrix A provides with the eigenvalue. But before using the relation between the coefficient and Matrices A and B, the relation was tested with basic matrices for which the answers were known. This was able to provide a check to make sure that the relation works and provides the correct answer when the required coefficients were substituted in them. Also after using the coefficients, the values were double checked by hand to make sure no mistakes were done while inputting the data.

7.2.6. Parameter Check

Throughout coding each part of the code, there was a check done for each value that was exported from the flight test data folder to the Python code. As the units of the data provided had to be changed for the required inputs, each time the data was put into the code, it was made sure that the values gotten made sense. For instance, if the height or the fuel weight made sense when the values were substituted in the required program. This ensured that no silly mistakes were taken and didn't produce any mistakes in further calculations.

7.3. System Testing

27

7.3.1. Stationary Measurements

A system test is conducted to be ascertain about the outputs that are acquired from the test series. Plots were constructed for both the 1^{st} and 2^{nd} series. These plots are meant to display certain trends (ex. linear, parabolic, etc.) and shapes and this is checked for. If they are not in agreement with expectations, the program is debugged and run again. Also, certain constants from the graphs have acceptable ranges for which they can be verified.

7.3.2. Dynamic Measurements

A system test is performed to ensure that the numerical output is within the acceptable range of 10% of the analytical model. The outputs used for testing are the eigenvalues of each maneuver. To system test the numerical model with the analytical model, the times of each maneuver are used as inputs to the models to ensure they are occurring at the same time instant and maneuver. The percentage error is calculated by finding the difference between the real and imaginary part of the eigenvalue, this is done as both parts of the eigenvalue affect different characteristics. Reproducing these results (also found in Section 4.3 and Section 5.3) into Table 7.1, show that most of the numerical results fall within the 10% margin set. The two eigenvalues that have a greater error than 10% are the real parts of both the phugoid and the Dutch roll. With the phugoid showing the largest difference, 36.754%, this was likely due to a modelling error in the

numerical model. These errors were dealt with in Section 8.5, by tweaking the stability coefficients such that the numerical results match the expected output. Besides the two outliers all the other eigenvalues fall within the 10% requirement for verification.

Table 7.1: Comparison and error between the analytical and numerical eigenvalues

Eigenmotion	Analytical	Numerical	Difference - *Real*	Difference - *Imaginary*
Short Period	-1.552 ± 2.460j	-1.552 ± 2.453j	0.030 %	0.253 %
Phugoid	-0.006 ± 0.134j	-0.004 ± 0.135j	36.754 %	1.124 %
Aperiodic Roll	-5.049	-5.136	1.7081 %	-
Dutch Roll	-0.449 ± 2.410j	-0.393 ± 2.499j	12.375 %	3.693 %
Aperiodic Spiral	0.011	0.0103	-1.455 %	-

28

Chapter 8

Validation

This chapter described the validation process for the simulation model by comparing the computational model results to the results obtained from the flight test.

8.1. Validation Experiment

The validation process requires a suitable validation experiment to be performed, one that can reliably determine whether the simulation accurately represents the physical problem. In this investigation, the validation experiment chosen was the flight test described in Chapter 3; the experimental results obtained from the eigenmode measurements were used to validate the simulation results.

8.2. Comparison of Eigenvalues and Errors

Here, the eigenvalues from the motions of the aircraft are derived from the flight data. The real and imaginary components (Equation 8.1 and Equation 8.2 respectively) are found by editing the equations in Section 4.3. The eigenvalues extracted from the flight data are presented and compared to the numerical ones in Table 8.1. As can be seen from the table there appears to be much larger errors than initially expected for this model. Regarding the periodic motions, the phugoid and dutch roll eigenvalues are relatively easy to extract from the data, however as short period is highly damped there is no completed oscillation. The eigenvalue for short period is estimated roughly from the data that is provided and the error of 64% is most likely due to the estimation. By analysing the graphs in Section 8.3, the error percentages are reasonable when also taking into account that the extraction of the eigenvalues is a process with unaccountable errors in itself. As the roll and spiral motion considered are aperiodic it was very difficult to accurately determine the eigenvalues from the flight data. Instead, these motions will be compared qualitatively based on the response plots from both the numerical and flight data, for this the reader is referred to Subsection 8.3.3 and Subsection 8.3.5.

29

$$\varepsilon_c = \frac{ln\frac{n_1}{n2}}{T_{n_1/n_2}} \qquad (8.1) \qquad\qquad \eta_c = \frac{2\pi}{P} \qquad (8.2)$$

Table 8.1: Comparison and error between the numerical and flight data eigenvalues

Eigenmotion	Numerical	Flight Data	Difference - *Real*	Difference - *Imaginary*
Short Period	-1.552 ± 2.453j	± 1.495	%	63.9 %
Phugoid	-0.0043 ± 0.135j	-0.0037 ± 0.143	16.2 %	5.7 %
Dutch Roll	-0.393 ± 2.499j	-0.455 ± 2.244	13.6 %	11.4 %

30

8.3. Simulated Responses

Now that a computational model has been made, it can be used to plot system or aircraft responses to certain control deflection inputs. When some specific control deflection inputs are acting on the aircraft, eigenmodes are be induced, which reflect the stability of the aircraft. These responses with respective control deflection inputs are simulated and are shown in Figures 8.1, 8.2, 8.3, 8.4, and 8.5.

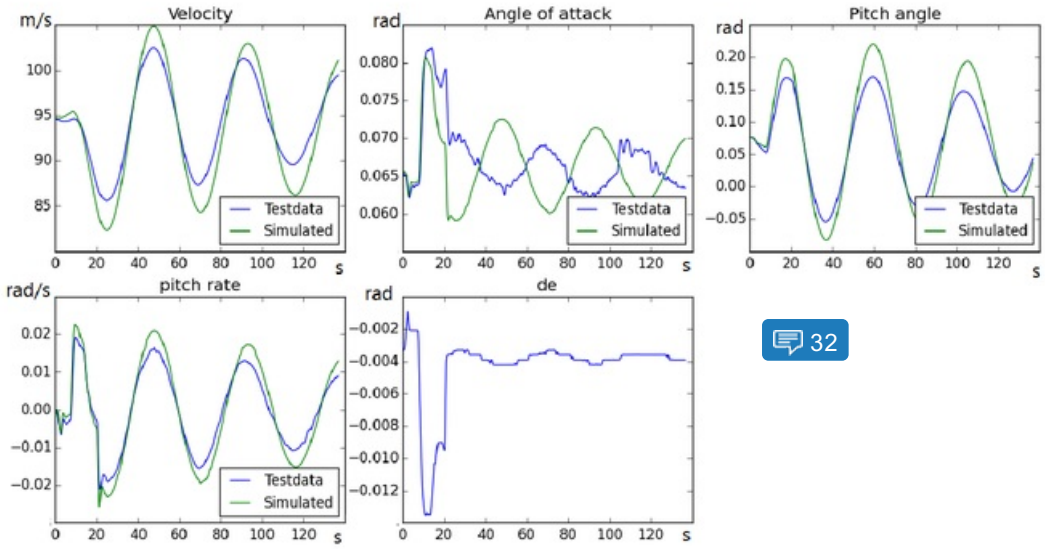
Next to the simulated response, also the measured responses are plotted as to compare and check for discrepancies. These responses and discrepancies will be discussed per graph. Besides control deflection input responses, also the aircraft or system response to a disturbance input should be considered. These show the response of the aircraft during steady straight flight after e.g. a sudden gust coming from below the aircraft causes a positive change in angle of attack. This example of disturbance input response is shown in Subsection 8.3.6.

8.3.1. Phugoid

The phugoid is one of the symmetric eigenmodes of the aircraft. It is symmetric because there is are only longitudinal motions happening, because of this only velocity represented by true airspeed, angle of attack, pitch angle, pitch rate and elevator deflection as input are relevant. To induce the phugoid motion, any kind of significant deflection change of the elevator can suffice, followed by keeping the deflection constant. As mentioned before, the phugoid is characterised by a large period, little dampened oscillation in velocity and pitch angle, while the angle of attack stays relatively constant, this can be seen back in Figure 8.1.

When comparing the simulated response to the measured test data, one can see that the oscillation frequency of all response outputs is very close the the real values, however, the amplitude is too large for all outputs. This could be to owe to effects described in Subsection 8.4.1. However, when looking at the angle of attack response, one can clearly see the simulated response having a 180°phase shift. This cannot be explained easily, but given the amplitude of the angle of attack oscillations, the relative effect due to these differences in angle of attack can be neglected. In effect, the angle of attack can be assumed invariable, even when plots shown a small oscillation.

31



32

Figure 8.1: Phugoid Motion

33

8.3.2. Short Period

The other symmetric eigenmode of the aircraft is the short period. To induce the short period, the pilot gives a short elevator control deflection impulse. The motion is highly dampened, but also highly periodic, which can be seen back in Figure 8.2.

When comparing the simulated response to the measured data, the discrepancies are fairly small. These

can, again, be explained by effects mentioned in Subsection 8.4.1, but are relatively small for most responses except for the angle of attack. The angle of attack has a similar response, but after the initial oscillation, the simulated response starts deviating downwards, away from the measured response. This could be to owe to the large changes in angle of attack over a small time period, magnifying the discretisation error.

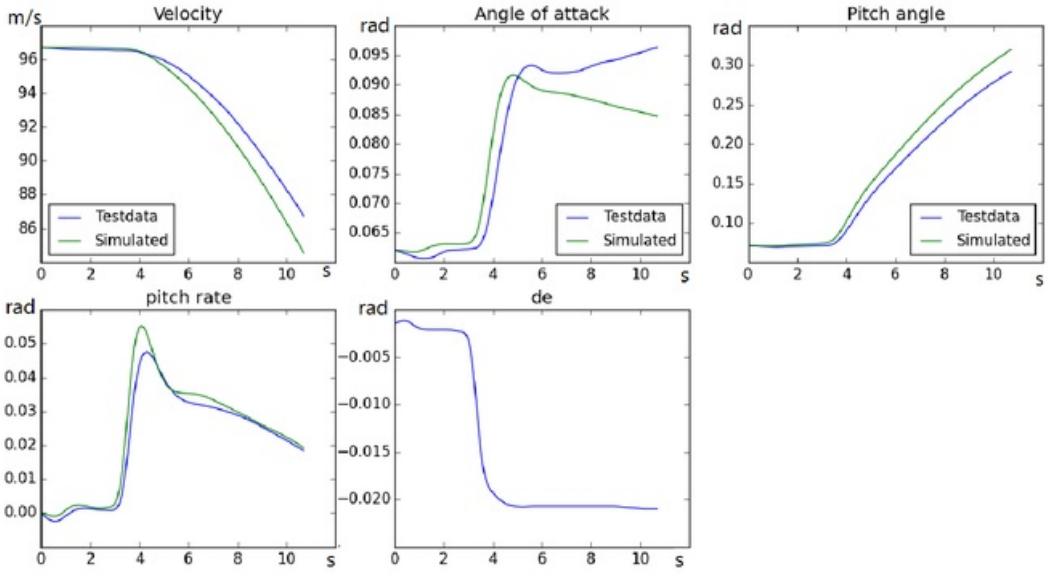


Figure 8.2: Short Period motion. Note that the motion pertains to...

8.3.3. Aperiodic Roll

The aperiodic roll is the first asymmetrical eigenmode considered for validation. This motion is called asymmetrical since it involves both rolling and yawing motions, making any motions performed by the aircraft asymmetrical. The aperiodic roll eigenmode control deflection inputs and aircraft responses can be seen in Figure 8.3. This motion specifically is induced by giving a step aileron deflection input. The motion is characterised by a quickly dampened out increase in roll rate, which remains constant after initial dampening. This almost constant roll rate ensures a near linear roll angle increase. It should be noted that this manoeuvre has no period, and is thus aperiodic, therefrom the name. It should also be noted that the yaw rate will start increasing, even though the rudder deflection remains relatively constant throughout the whole motion. This can be explained due to the effective local angle of attack increase over one wing while rolling, inducing more drag on this one wing, thus dragging it backwards, which in turn creates a yawing motion.

When comparing the simulated response to the measured data, the roll angle and roll rate have fairly small discrepancies. The roll rate has some initial differences, but this can be to owe to time frame selections: it could be that the aircraft was not fully steady before the manoeuvre started, which is one of the assumptions made for creating the simulated response. This same effect can be reflected on the yaw rate, however, here the initial discrepancy further effects the rest of the response by shifting the simulated yaw rate plot upwards. The shape of the response after initial discrepancy however, remains similar to the measured response. It should be noted that apparent oscillations in rudder deflection can be neglected due to the amplitude of these oscillations being insignificant. This can be to owe to sensor accuracy.

8.3.4. Dutch Roll

The second of the asymmetric eigenmodes considered is the dutch roll. This motion can be seen in Figure 8.1. This motion can be induced by giving an impulse rudder deflection and is characterised by a highly periodic, dampened response in roll angle, roll rate and yaw rate. It should be noted that also a small change in deflection for aileron can be seen, but the magnitude of this deflection is negligibly small and can be blamed on the ailerons reacting to sudden changes in roll rate.

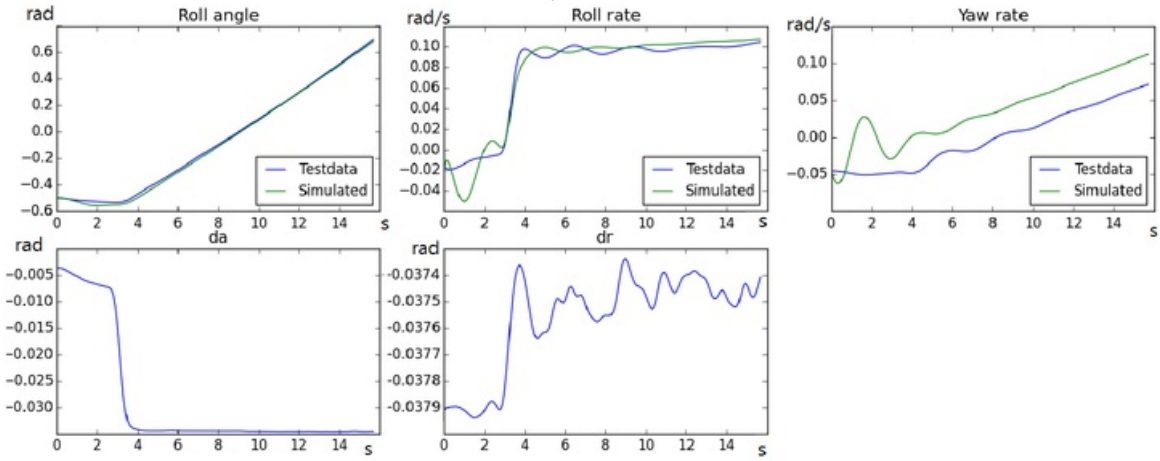


Figure 8.3: Aperiodic Roll Motion

When comparing the simulated response to the measured data, it can be seen that the roll angle, roll rate and yaw rate all have amplitudes relatively close to the measured data. However, it can also be seen that the period of these oscillations is slightly too small, but this effect is small and can be due to effects explained in Section 8.4.1. Besides this the plots seem to converge relatively well without further adjustments.

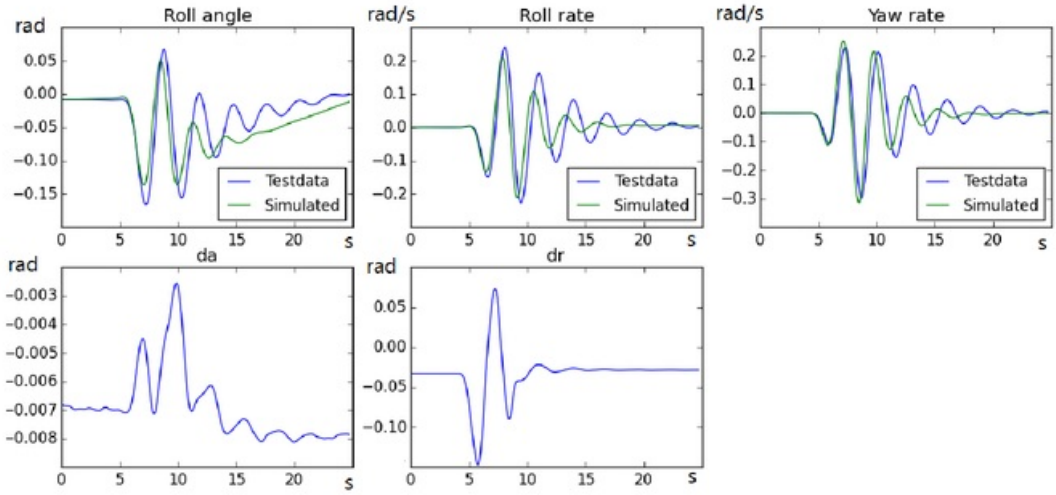


Figure 8.4: Dutch Roll Motion

8.3.5. Spiral

The last of the asymmetric eigenmodes is the spiral. This motion can be seen inFigure 8.5. The spiral can be induced by an impulse aileron deflection, and can be observed over a longer period of time (100+ seconds). This is the only unstable eigenmode of the aircraft considered for this assignment and is characterised by a small roll angle, increasing non linearly due to a small increasing roll rate. Together with the roll rate, the yaw rate also increases due to effects explained in Subsection 4.2.1.

When comparing the simulated response to the measured data, it can be seen that the roll angle, roll rate and yaw rate all match the measured data relatively well. Any small discrepancies over time can be explained by effects mentioned in Subsection 8.4.1. It should also be mentioned that apparent oscillations in rudder deflection are negligible due to insignificant amplitudes. This can be to owe to sensor accuracy.

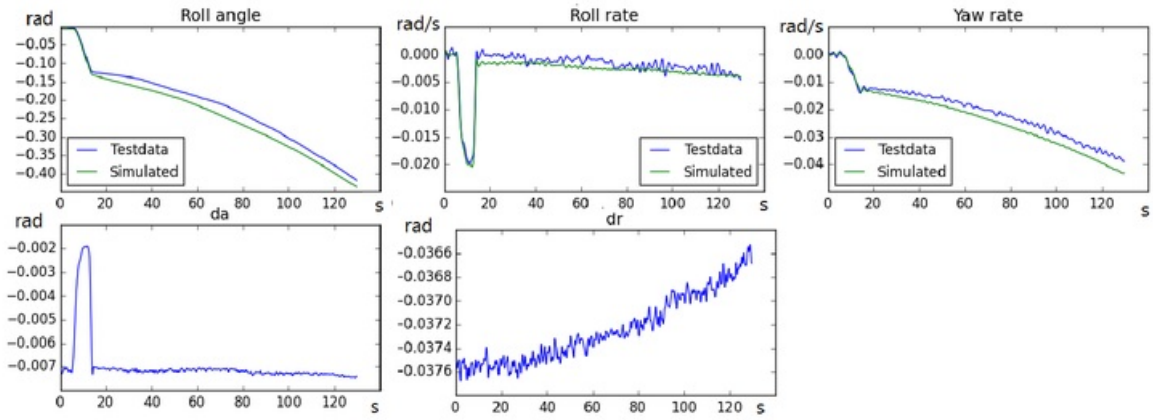


Figure 8.5: Spiral Motion

8.3.6. Response to Disturbance Input

The system response to a disturbance input is equally important as the eigenmodes to determine the aircraft stability. The chosen disturbance input to discuss is a gust coming from below the aircraft. This is an easy example to compare, since it will only result in a symmetrical response, thus not requiring roll and yaw comparison. Using the same method as explained below, other disturbance inputs can be evaluated as well.

A good estimate for gust winds is about 16 knots or 28.8m/s.[17] When taking the average aircraft true airspeed of 112m/s, this will effectively induce a sudden change in angle of attack of $arctan(\frac{28.8}{112}) = 0.251$ rad. To compare the undisturbed aircraft response with a disturbed motion, a steady horizontal flight section is extracted from the measured data as to obtain aircraft trim conditions. The undisturbed motion is then the response of the aircraft to these initial conditions, while the disturbed response is the motion of the aircraft when the initial angle of attack is increased by 0.251rad from the trim conditions.

The undisturbed response is plotted in Figure 8.6, while the disturbed response is plotted in Figure 8.7. When looking at the undisturbed response, one can notice some initial oscillations in angle of attack and pitch rate. These oscillations can be to owe to slight deviations between initial conditions and perfectly trimmed initial conditions. Besides this, also velocity, angle of attack, pitch angle and pitch rate are oscillating. This can be explained as some kind of phugoid, while the aircraft is slowly returning to steady straight flight through very lightly dampened oscillations. However, these oscillations are relatively small and considering the relatively large time scale, they can be said to be insignificant.

The simulated disturbance aircraft response is plotted in Figure 8.7. One can see that the initial oscillations in pitch rate and angle of attack also occur here, however, this time they have a much larger magnitude. This is the aircraft responding to the large difference between trimmed conditions and initial conditions, although these are quickly dampened out. The larger period oscillations in velocity, angle of attack, pitch angle and pitch rate are also reoccurring, and can be found to have a much larger amplitude than for the undisturbed response. This can also be explained as some kind of phugoid, while the aircraft is slowly returning to steady straight flight through very lightly dampened oscillations.

8.4. Uncertainty Assessment

The flight test only investigated the response to control inputs and not to disturbance inputs. As such, the experimental results were compared with the simulation results only pertaining to the control input response (Section 8.3). Errors introduced due to modelling are assessed in Subsection 8.4.1 while input data errors are briefly examined in Subsection 8.4.2. In Subsection 8.4.3, the theory used in the equations of this simulation is briefly looked at.

8.4.1. Modelling Errors

The errors previously ascertained can primarily be attributed to the transition from the physical problem to the physical model, namely modelling errors. These errors are discussed here. There were some assumptions that produced only very minor errors. For example, assuming gravity is constant with altitude results in only a

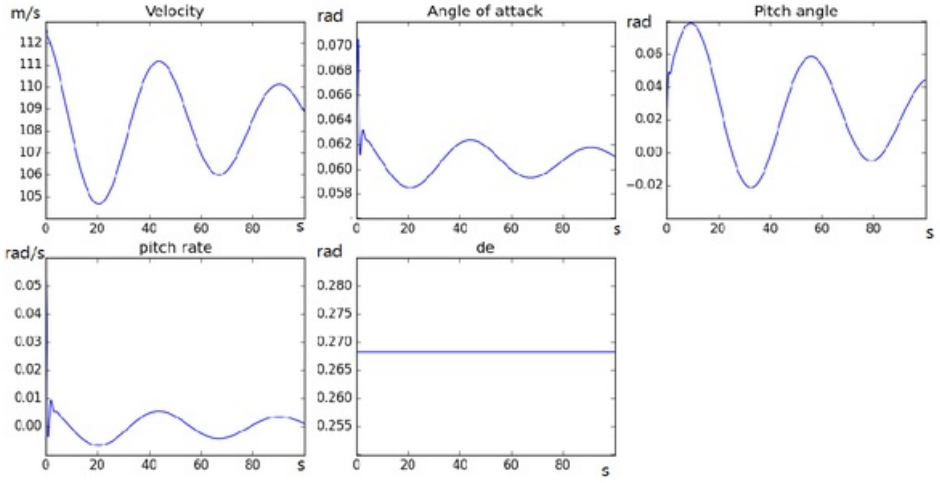


Figure 8.6: Aircraft Response to Undisturbed Steady Straight Flight

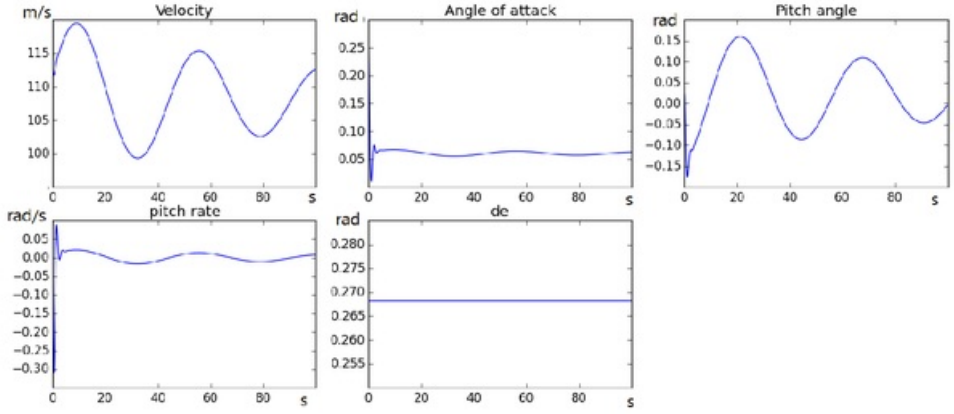


Figure 8.7: Aircraft Response to Disturbance Input

1% error at 32 km altitude [5]. However, other errors had a more significant impact on the simulation results.

The Earth was assumed spherical, thereby leaving only the Kepler central force term [8]. Compared with full model of gravitational potential, Legendre terms included, the results should only have minute discrepancies due to the fact that the experimental data was gathered over a small latitudinal range.

The flat Earth assumption ignored the curvature of the Earth, thereby ignoring any rotation of the E-frame with respect to the ECI frame. The flight was carried out in a very narrow latitudinal range; since it was almost constant, the omitted rotation can be estimated in order to gauge the impact of the assumption. Equation 8.3 gives the distance travelled moving along a latitude circle for a certain change in longitude, where r is the radial distance from the centre of the Earth (using spherical assumption); and τ and δ are in radians [18].

$$\text{Distance} = \int_0^{\Delta\tau} r\cos(\delta)\,d\tau = \Delta\tau\, r\cos(\delta) \tag{8.3}$$

The test aircraft travelled back and forth between Rotterdam and Nijmegen, an aerial distance of approximately 90 km, at about 2 km altitude. The average latitude was 51.87°on this route [19]. Substituting these values yields $\Delta\tau = 1.31°$. This is equivalent to the angle that the E-frame rotates through along the *entire* route, which amounts to a 0.36% error in its orientation, using the flat Earth assumption. Hence, the corresponding error for any of the eigenmodes, performed over just a fraction of the route, is negligible.

Wind was neglected in the simulation; however, it was obviously present during the flight test. For the relatively short distances involved in the simulation, only high wind shears would have a noticeable impact on the motion of the aircraft. One common indicator of wind shear is a sudden temperature change of more than 5°C at a certain altitude; the experimental data did not indicate such a phenomenon. Furthermore, due to a combination of the flat nature of the Netherlands; the congenial weather and lack of storms on the day; and the relatively low altitude (5000-8000 ft), it can safely be considered that wind had a negligible effect on the kinematic velocity [20].

The assumption that the Earth is non-rotating can yield large errors over large timespans (order of hours); however, for the relatively short timespans (order of seconds) in this investigation, errors are much reduced and thus have little effect on the results.

The aircraft mass was assumed to be constant during each simulated eigenmode. During the flight test, the mass was obviously continuously decreasing due to fuel consumption. As such, for longer eigenmotions, larger state errors occur due to accumulated error in the mass and centre of gravity; additionally, the damping becomes overestimated with constant mass. Furthermore, the actual aircraft experiences elastic modes during flight, and is not perfectly rigid. Although this affects the results, the normal flight conditions of interest do not involve high elastic deformations [5].

8.4.2. Data Errors

The equipment on board that was used to record the relevant parameters during the flight test, introduces its own errors. In addition, the data that was acquired for the horizontal stationary flight was recorded by hand while looking at the equipment on board. These may have caused a data error too as there might be a mistake while noting down the values. Also, there was some noise in the graphs that were produced. This maybe because of the turbulence due to the weather or maybe due to the equipment too.

8.4.3. Discussion of Theory Used

The entire model for the behaviour of the aircraft is constructed on the linearised equations of motion and under the assumption that the different motions start when the aircraft is in horizontal, steady flight. However, in reality the initial condition could be non-linear, meaning that second order and higher stability derivatives will in fact affect the response of the aircraft and cannot be neglected. This could be accounted for by constructing a non-linearised model or by constructing linearised equations around different initial conditions. For the purposes of this report however linearised equations of motion around the initial steady state, horizontal flight condition is considered valid as this was also performed during the test flight. Moreover, by using the linearised equations of motion the characteristic modes of the aircraft can be analyzed. Furthermore, the theory utilised does not take into account any change in weights during the motions considered. For certain motions, such as short period, the effect of this is negligible due to the fact that the motion lasts only a few seconds. For the phugoid or spiral motion, however, the motion lasts significantly longer and thus the change in weight may impact the result.

8.5. Key Stability Derivatives Tweaking

During creation of the numerical simulation model, some discrepancies between the simulated aircraft response and the measured aircraft response can be seen when these are plotted, as seen in Figures 8.1, 8.2, 8.3, 8.4 and 8.5. Some of these discrepancies, however, can be mitigated by changing some of the stability derivatives, from which the simulated response depends. As determined in Section 4.2 and explained in Subsection 6.1.4, some stability derivatives have much larger effects on the eigenmode than others; these are the key stability derivatives. By tweaking these, the simulated responses can be improved to match the measured responses, and thus creates a more realistic simulator. These coefficients are multiplied by some tweaking factor as to match the measured test data as closely as possible, which are presented in Table 8.2. The improved simulated responses are then plotted, and shown in Figures 8.8, 8.9, 8.10, 8.11 and 8.12. It should be noted that the short period still deviates significantly from the measured data, however this mode could not be improved more using the tweaking factors due to phugoid codependency on the same key derivatives.

Table 8.2: Key stability derivatives and tweaking factors

Dependent Motions	Key Stability Derivatives	Value	Tweaking factor
Short Period+Phugoid	C_{Z_α}	-5.74340	0.74
Short Period+Phugoid	C_{m_q}	-8.79415	0.68
Aperiodic Roll	C_{l_p}	-0.71085	0.98
Dutch Roll+Spiral	C_{n_β}	0.1348	0.88
Dutch Roll+Spiral	C_{n_r}	-0.2061	0.91

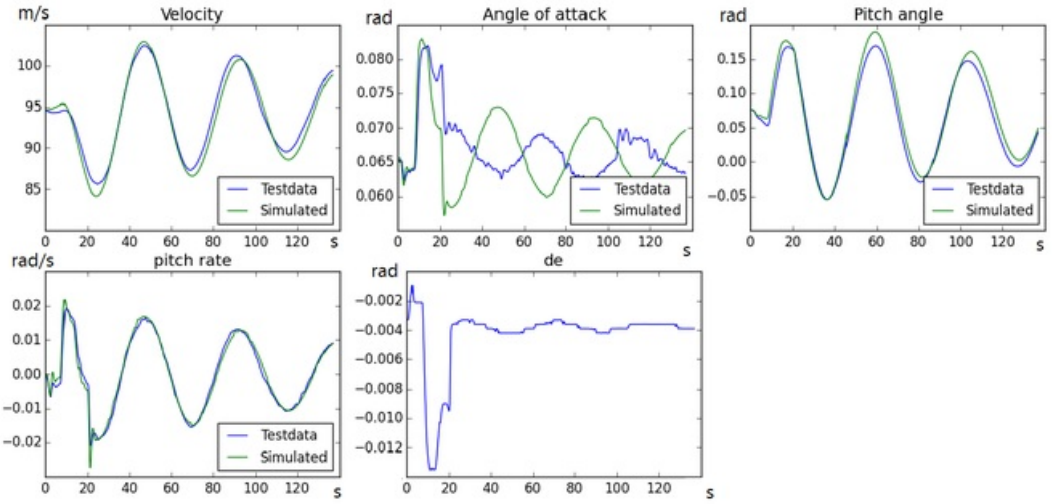


Figure 8.8: Phugoid Motion

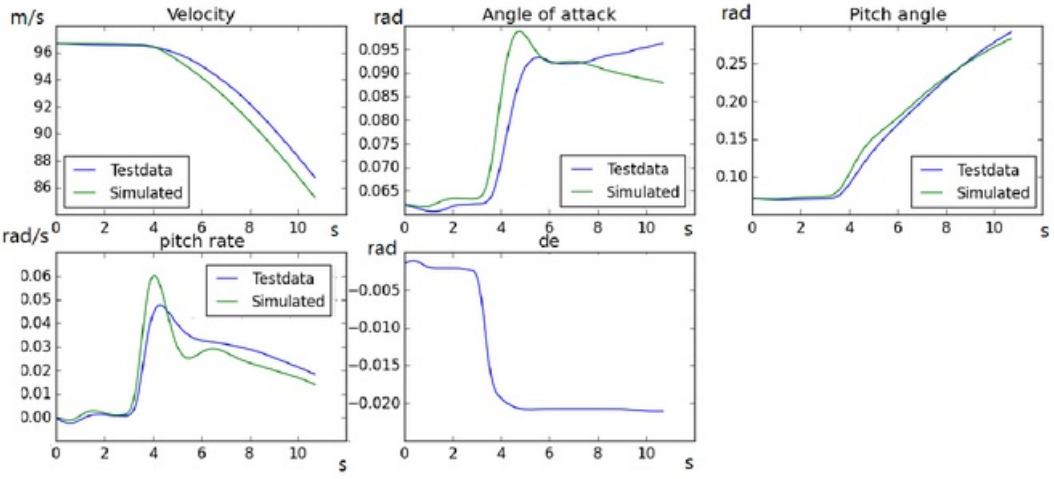


Figure 8.9: Short Period Motion

8.6. Recommendations

To improve the model, several recommendations are listed below.

- To get more accurate results and a graph for each of the manoeuvre, more and longer trials should be taken. Getting more points would help to make it easier to compare the data.

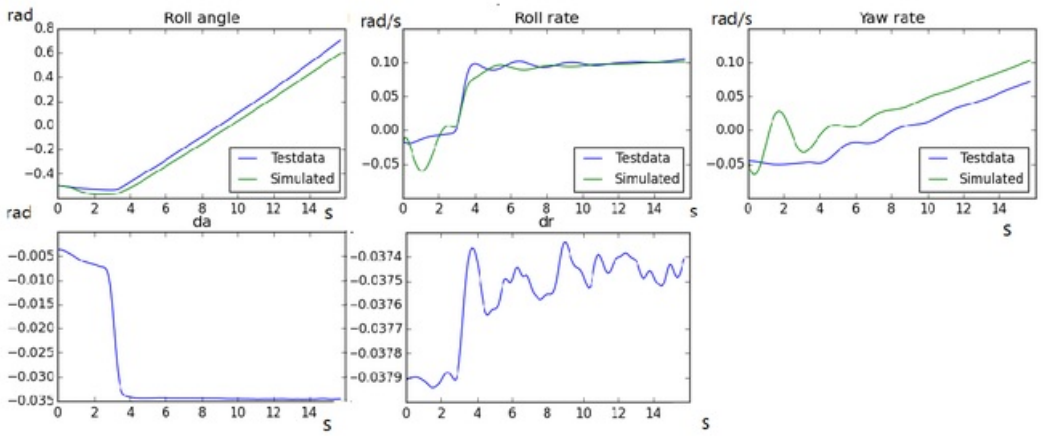


Figure 8.10: Aperiodic Roll Motion

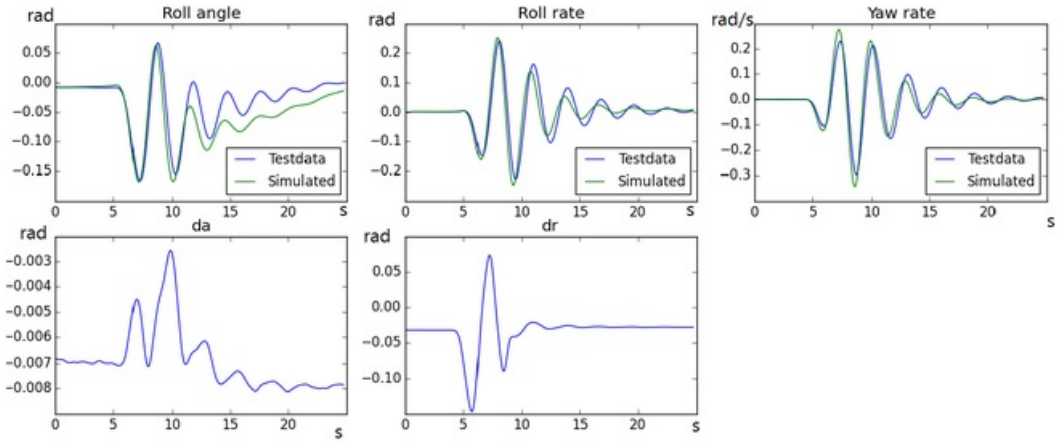


Figure 8.11: Dutch Roll Motion

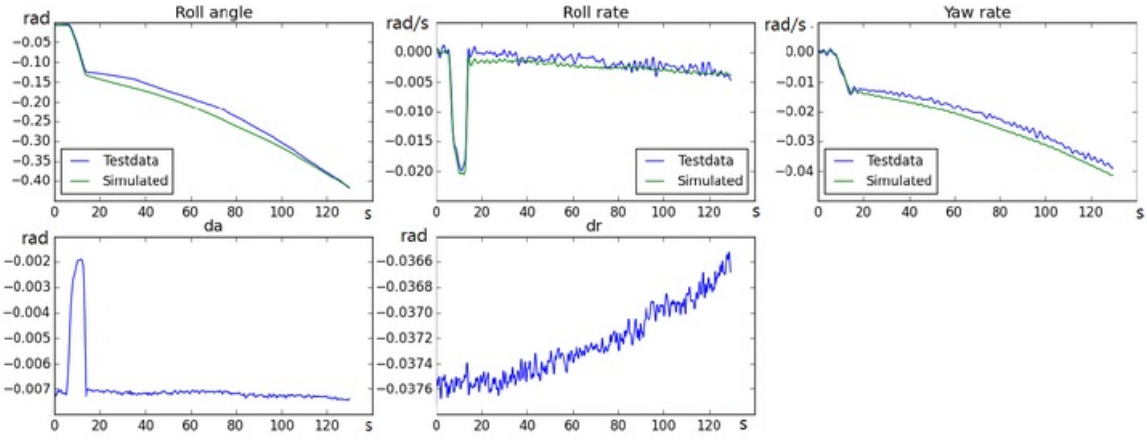


Figure 8.12: Spiral Motion

- Instead of linearisation, the equations of motion could be kept in their non-linear form and a numerical integration performed in order to realise a more accurate simulation of the aircraft's response. This would also allow the model to better simulate different initial conditions other than the initial horizontal, steady state flight currently used.

37

- Perform test flight at higher altitude (business jets typically fly at FL400). Aerodynamic forces are lower due to lower air density; as such, damping is decreased. It is important to investigate this effect as this could provide valuable practical insight into the aircraft performance during regular operations.

- When considering the disturbance input, only a gust wind from below the aircraft was considered, resulting in a symmetric motion of the aircraft. In reality, the aircraft could experience a variety of different disturbance inputs such as a gust causing a yawing or rolling motion, which should be investigated.

Chapter 9

Conclusion

The goal of this team was to design a computational model that simulates the response in the various states of a new business jet to certain control inputs and disturbances; this pertains to the aircraft's characteristic modes, or 'eigenmotions', which are independent of the input provided. A thorough verification and validation process was executed in order to establish the reliability of the results produced and, more importantly, the suitability of the simulation tool to the future development stages of the aircraft. Without such a process, the development could well run into later issues that precipitate exceedingly expensive design modifications.

The project began with setting out a problem approach where the problem, as it stood in the real world, was transformed into something that could realistically be tackled in a computational manner. This involved applying a set of simplifying assumptions that paved the way to building a mathematical description of this real-world problem. A prototype flight test gathered relevant data for helping to build models of the aircraft motion, out of this mathematical description. Firstly, an analytical model was built, which utilised supplementary assumptions to simplify the problem further and allow closed-form equations to be solved. Secondly, a numerical model was built whereby the equations of motion of the aircraft were rewritten into a state-space form; this allowed a discretisation to be performed and the formulation of a numerical algorithm to implement the model computationally. This led to the creation of the simulation program itself, which was coded in the Python language. Verification involved unit, integration, and system testing of the code; these tests occurred on varying timelines throughout the project duration. Equipped with a verified set of simulation results the model was compared to data obtained from the flight test. Once the numerical model was validated, it was further optimised by altering the stability coefficients to match the numerical model to the flight data. In the future it is recommended to perform a test flight with a longer duration to get the best possible results for the long period and spiral as they have a relatively long duration. Also, the model should be tested against a larger range of disturbance inputs to see the response for each.

Bibliography

[1] D.P. Raymer. *Aircraft Design: A Conceptual Approach.* American Institute of Aeronautics and Astronautics, Washington DC, USA, 2nd edition, 1992.

[2] M. Cook. *Flight Dynamics Principles.* Butterworth-Heinemann, London, UK, 2nd edition, 2007.

[3] E. Mooij, Z. Papp, and W. van der Wal. *AE3212-II Simulation, Verification and Validation Lecture Notes.* Faculty of Aerospace Engineering, Delft University of Technology, February 2017.

[4] W.W. Royce. *Managing the Development of Large Software Systems.* California Institute of Technology, 1970.

[5] J.A. Mulder, W.H.J.J. van Staveren, J.C. van der Vaart, E. de Weerdt, C.C. de Visser, A.C. in't Veld, and E. Mooij. *AE3212-I Flight Dynamics Lecture Notes.* Faculty of Aerospace Engineering, Delft University of Technology, March 2013.

[6] A.C. in't Veld, M. Naeije, and E. Mooij. *AE3212-I Flight Dynamics Lecture Series.* Faculty of Aerospace Engineering, Delft University of Technology, 2017.
[7] Airliners. *Mid-sized Corporate Jets*, 2016. `http://www.airliners.net/forum/viewtopic.php?t=69799` (Accessed: 22/03/17).
[8] R. Noomen. *AE2230-I Orbital Mechanics Lecture Series.* Faculty of Aerospace Engineering, Delft University of Technology, 2016.
[9] H.F.R Schöyer, J.W. Cornelisse, and K.F. Wakker. *Rocket Propulsion and Spaceflight Dynamics.* Pitman, London, UK, 1st edition, 1979.
[10] J. Roskam. *Airplane Flight Dynamics and Automatic Flight Controls.* DARcorporation, Lawrence, USA, 1st edition, 1995.
[11] J.D. Anderson. *Fundamentals of Aerodynamics.* McGraw-Hill, New York, USA, 5th edition, 2011.
[12] G.J.J. Ruijgrok. *Elements of Airplane Performance.* VSSD, Delft, The Netherlands, 2nd edition, 2007.
[13] M. Voskuijl. *AE2230-I Flight Mechanics Lecture Series.* Faculty of Aerospace Engineering, Delft University of Technology, 2016.
[14] E. Gill, G. La Rocca, W. Verhagen, and A. Cervone. *AE3211-I Systems Engineering & Aircraft Design Lecture Series.* Faculty of Aerospace Engineering, Delft University of Technology, February 2017.
[15] A.C. in't Veld and T.J. Mulder. *AE3212-II SVV Flight Dynamics Assignment.* Faculty of Aerospace Engineering, Delft University of Technology, January 2017.
[16] W. van der Wal. *AE3212-II SVV Introduction Lecture.* Faculty of Aerospace Engineering, Delft University of Technology, February 2017.
[17] National Weather Service Corporate Image Web Team. *NOAA's National Weather Service - Graphical Forecast*, 2017. `https://graphical.weather.gov/` (Accessed: 29/03/17).
[18] R. Noomen. *AE1110-II Introduction to Aerospace Engineering I Lecture Series.* Faculty of Aerospace Engineering, Delft University of Technology, 2016.
[19] Google Maps. *Map of The Netherlands*, 2017. `https://www.google.nl/maps/place/Netherlands` (Accessed: 28/03/17).
[20] Federal Aviation Administration. *Wind Shear*, 2008. `https://www.faasafety.gov` (Accessed: 21/03/17).

Appendix A

Task Distribution

In this appendix, the task distribution table is given. The names refer to the group members working on each section and subsection of this report. The overview per chapter is given in Table A.1.

Table A.1: Task Distribution

	Student	Ankit	Bas	Sumalik	Thomas	Kiko	Alex
Task	Time (hours)						
Introduction	6.5	0.5	0	0	0.5	0.5	4
Problem Analysis	57	0.5	1	20	20	0.5	15
Experimental Data Analysis	63	0.5	0	30	12	0.5	10
Analytical Model	62	2	30	0	0	30	0
Numerical Model	35	20	0	0	10	5	0
Computational Model	56	20	0	4	30	2	0
Verification	41	8	8	5	0	5	10
Validation	62	2	15	0	20	5	10
Conclusion	19	1	1	2	10	0	5
Reporting/Checking	117	25	20	15	12	25	45
Total Time	**518.5**	**79.5**	**75**	**76**	**114.5**	**73.5**	**99**

38

Appendix B

Simulation Code

In this appendix, the programming code, written in the Python language is provided.

Flight Data Reader: 'flightdata.py'

```
import scipy.io
import numpy as np
flightdata = scipy.io.loadmat('flightdata.mat')['flightdata']

#environmental data
h = np.transpose(flightdata['Dadc1_alt'][0][0][0][0][0])[0]*0.3048
#altitude m
T = np.transpose(flightdata['Dadc1_sat'][0][0][0][0][0])[0] +273.15
#temperature
t = (flightdata['time'][0][0][0][0][0])[0]
#timestamp in sec
V = np.transpose(flightdata['Dadc1_tas'][0][0][0][0][0])[0]*0.5144      #m/s

#input data
de = np.transpose(flightdata['delta_e'][0][0][0][0][0])[0]*np.pi/180.      #elevator deflection (rad)
da = np.transpose(flightdata['delta_a'][0][0][0][0][0])[0]*np.pi/180.      #aileron deflection (rad)
dr = np.transpose(flightdata['delta_r'][0][0][0][0][0])[0] *np.pi/180.     #rudder deflection (rad)

#reaction data
p = np.transpose(flightdata['Ahrs1_bRollRate'][0][0][0][0][0])[0]*np.pi/180.
#roll rate of the aircraft (rad/s)
q = np.transpose(flightdata['Ahrs1_bPitchRate'][0][0][0][0][0])[0]*np.pi/180.
#pitch rate (rad/s)
r = np.transpose(flightdata['Ahrs1_bYawRate'][0][0][0][0][0])[0]*np.pi/180.
#yaw rate (rad/s)
theta = np.transpose(flightdata['Ahrs1_Pitch'][0][0][0][0][0])[0]*np.pi/180.
#pitch angle (rad)
phi = np.transpose(flightdata['Ahrs1_Roll'][0][0][0][0][0])[0]*np.pi/180.
#roll angle (rad)
alpha = np.transpose(flightdata['vane_AOA'][0][0][0][0][0])[0]*np.pi/180.
#aoa (rad)

#fuel stuff
ffl = np.transpose(flightdata['lh_engine_FMF'][0][0][0][0][0])[0]*0.000125997881
#fuel flow at left engine kg/s
ffr = np.transpose(flightdata['rh_engine_FMF'][0][0][0][0][0])[0]*0.000125997881
#fuel flow at right engine kg/s

#fuel used at any point
fused = np.zeros(len(ffr))
fused[0]=0.
for i in xrange(1,len(ffr-1)):
    fused[i] = fused[i-1]+(t[i]-t[i-1])*(ffl[i]+ffr[i])
```

```
#GPS data for plotting cool stuff
gpsx = np.transpose(flightdata['Gps_lat'][0][0][0][0][0])[0]       # gps latitude
gpsy = np.transpose(flightdata['Gps_long'][0][0][0][0][0])[0]      # gps longitude
gpssec = np.transpose(flightdata['Gps_utcSec'][0][0][0][0][0])[0] # gps time stamp in sec

def inp():
    return [t,h,T,V,de,da,dr,p,q,r,theta,phi,alpha,fused]
```

Response Simulation: 'simulator.py'

```
import numpy as np
import control.matlab as cm
import scipy.linalg as la
from matplotlib import pyplot as plt
import flightdata

#importing data from the flight test for input and comparison

data = flightdata.inp() #full lists, non filtered for manoevres -> t'full'
tf   = data[0] #time (s)
hf   = data[1] #pressure altitude  (m)
Tf   = data[2] #temperature (K)
Vf   = data[3] #velocity (m/s)
deff = data[4] #elevator deflection (rad)
daf  = data[5] #aileron deflection (rad)
drf  = data[6] #rudder deflection (rad)
pf   = data[7] #roll rate (rad/s)
qf   = data[8] #pitch rate (rad/s)
rf   = data[9] #yaw rate (rad/s)
thetaf = data[10] #pitch angle (rad)
phif   = data[11] #roll angle (rad)
alphaf = data[12] #angle of attack (rad)
fusedf = data[13] #fuel used (kg)

#recognising which manoevre is dutch roll and aperiodic roll
dutch = []
for i in xrange(len(tf)):
    if tf[i]>2622. and tf[i]<2647.:
        dutch.append(i)

aperiodic = []
for i in xrange(len(tf)):
    if tf[i]>2991. and tf[i]<3007.:
        aperiodic.append(i)

#fixed data that does not have to be recalculated per manoevre

OEW = 9165./2.2
fuelmass = 2700./2.2
peoplemass = 99.+90.+99.+74.+66.+79.+80.+91.+84

# aerodynamic properties

e      = 0.727797           # Oswald factor [ ]
CD0    = 0.021008           # Zero lift drag coefficient [ ]
```

```
CLa     = 4.56              # Slope of CL-alpha curve [ ]

# Longitudinal stability
Cma     = -0.70779            # longitudinal stabilty [ ]
Cmde    = -1.4224             # elevator effectiveness [ ]

# Aircraft geometry

S       = 30.00           # wing area [m^2]
Sh      = 0.2 * S         # stabiliser area [m^2]
Sh_S    = Sh / S          # [ ]
lh      = 0.71 * 5.968    # tail length [m]
c       = 2.0569          # mean aerodynamic cord [m]
lh_c    = lh / c          # [ ]
b       = 15.911          # wing span [m]
bh      = 5.791           # stabilser span [m]
A       = b ** 2 / S      # wing aspect ratio [ ]
Ah      = bh ** 2 / Sh    # stabilser aspect ratio [ ]
Vh_V    = 1                  # [ ]
ih      = -2 * np.pi / 180   # stabiliser angle of incidence [rad]

# Constant values concerning atmosphere and gravity

rho0   = 1.2250          # air density at sea level [kg/m^3]
Lambda = -0.0065         # temperature gradient in ISA [K/m]
Temp0  = 288.15          # temperature at sea level in ISA [K]
R      = 287.05          # specific gas constant [m^2/sec^2K]
g      = 9.81            # [m/sec^2] (gravity constant)


# Aerodynamic constants

Cmac   = 0                      # Moment coefficient about the aerodynamic centre [ ]
CNwa   = CLa                    # Wing normal force slope [ ]
CNha   = 2 * np.pi * Ah / (Ah + 2) # Stabiliser normal force slope [ ]
depsda = 4 / (A + 2)            # Downwash gradient [ ]

#enabling simulating system for every manoevre

def simulate(time):     #'time' are the indexes of the time period in list t

    t  = tf[time[0]:time[-1]]   #time (s)
    h  = hf[time[0]:time[-1]]   #pressure altitude  (m)
    T  = Tf[time[0]:time[-1]]   #temperature (K)
    V  = Vf[time[0]:time[-1]]   #velocity (m/s)
    de = np.array(deff[time[0]:time[-1]])   #elevator deflection (rad)
    da = np.array(daf[time[0]:time[-1]])    #aileron deflection (rad)
    dr = np.array(drf[time[0]:time[-1]])    #rudder deflection (rad)
    de = de-de[0]       #correcting for initial deflection
    da = da-da[0]       #correcting for initial deflection
    if time==dutch:
        dr = -dr-dr[0]
    if time==aperiodic:
        dr = np.zeros(len(dr))-dr[0]
    else:
        dr = dr-dr[0]
```

```
p  = pf[time[0]:time[-1]]   #roll rate (rad/s)
q  = qf[time[0]:time[-1]]   #pitch rate (rad/s)
r  = rf[time[0]:time[-1]]   #yaw rate (rad/s)
theta = thetaf[time[0]:time[-1]]    #pitch angle (rad)
phi   = phif[time[0]:time[-1]]      #roll angle (rad)
alpha = alphaf[time[0]:time[-1]]    #angle of attack (rad)

# Stationary flight condition

hp0    = h[0]           # pressure altitude in the stationary flight condition [m]
V0     = V[0]           # true airspeed in the stationary flight condition [m/sec]
alpha0 = alpha[0]       # angle of attack in the stationary flight condition [rad]
th0    = theta[0]       # pitch angle in the stationary flight condition [rad]
q0 = q[0]
p0 = p[0]
r0 = r[0]
phi0 = phi[0]
fuelused = fusedf[time[0]] #fuel used (kg)

# Aircraft mass

m      = OEW+fuelmass-fuelused+peoplemass          # mass [kg]

# air density [kg/m^3]

rho    = rho0**( ((1+(Lambda * hp0 / Temp0)))/ (-((g / (Lambda*R)) + 1)))
W      = m * g             # [N]        (aircraft weight)

# Constant values concerning aircraft inertia

muc    = m / (rho * S * c)
mub    = m / (rho * S * b)
KX2    = 0.019
KZ2    = 0.042
KXZ    = 0.002
KY2    = 1.25 * 1.114

# Lift and drag coefficient
CL = 2 * W / (rho * V0 ** 2. * S)              # Lift coefficient [ ]
CD = CD0 + (CLa * alpha0) ** 2 / (np.pi * A * e) # Drag coefficient [ ]

# Stabiblity derivatives
# uncomment tweaking factors for more accurate results

CX0    = W * np.sin(th0) / (0.5 * rho * V0 ** 2 * S)
CXu    = -0.02792
CXa    = -0.47966
CXadot = +0.08330
CXq    = -0.28170
CXde   = -0.03728

CZ0    = -W * np.cos(th0) / (0.5 * rho * V0 ** 2 * S)
CZu    = -0.37616
CZa    = -5.74340 #*0.74#short period + phugoid
CZadot = -0.00350
```

```
CZq     = -5.66290
CZde    = -0.69612

Cmu     = +0.06990
Cmadot  = +0.17800
Cmq     = -8.79415 #*0.68#short period + phugoid

CYb     = -0.7500
CYbdot  =  0
CYp     = -0.0304
CYr     = +0.8495
CYda    = -0.0400
CYdr    = +0.2300

Clb     = -0.10260
Clp     = -0.71085 #*0.98#aperiodic roll
Clr     = +0.23760
Clda    = -0.23088
Cldr    = +0.03440

Cnb     =  +0.1348 #*0.88#dutch roll
Cnbdot  =   0
Cnp     =  -0.0602
Cnr     =  -0.2061 #*0.91#dutch roll
Cnda    =  -0.0120
Cndr    =  -0.0939

#asymmetrical system
AS_C1 = np.mat([[(CYbdot - 2*mub)*b/V0, 0, 0, 0],\
                [0, -0.5*b/V0, 0, 0],\
                [0, 0, -4*mub*KX2*b/V0, 4*mub*KXZ*b/V0],\
                [Cnbdot*b/V0, 0, 4*mub*KXZ*b/V0, -4*mub*KZ2*b/V0]])

AS_C2 = np.mat([[CYb, CL, CYp, CYr-4*mub],\
                [0, 0, 1, 0],\
                [Clb, 0, Clp, Clr],\
                [Cnb, 0, Cnp, Cnr]])

AS_C3 = np.mat([[CYda, CYdr],\
                [0, 0],\
                [Clda, Cldr],\
                [Cnda, Cndr]])

AS_A = -np.linalg.inv(AS_C1)*AS_C2
AS_B = (-np.linalg.inv(AS_C1)*AS_C3)
AS_C = np.identity((4))
AS_D = np.zeros((4,2))
ASYM = cm.ss(AS_A,AS_B,AS_C, AS_D)

#symmetrical system

C1 = np.matrix([[-2.*muc*c/V0 , 0 ,0,0],
                [0 ,(CZadot-2.*muc)*c/V0, 0 ,0],
                [0 ,0 ,-c/V0 ,0],
                [0 ,c/V0*Cmadot ,0 ,-2*muc*KY2*c/V0]])
```

```
C2 = np.matrix([[CXu, CXa, CZ0, CXq],
                [CZu, CZa, -CX0, CZq+2*muc],
                [0, 0, 0, 1],
                [Cmu, Cma, 0, Cmq]])

C3 = np.matrix([[CXde],
                [CZde],
                [0],
                [Cmde]])

S_A = -np.linalg.inv(C1) * C2
S_B = -np.linalg.inv(C1) * C3
S_C = np.identity(4)
S_D = np.zeros((4,1))
SYM = cm.ss(S_A,S_B,S_C,S_D)

#eigenvalues and natural frequencies

eigenvalue2 = np.array(la.eig(ASYM.A)[0])
eigenvalue1 = np.array(la.eig(SYM.A)[0])

# period of eigenmodes including amplitude

Thalf1 = np.log(0.5)*c/np.real(eigenvalue1)/V0 #amplitude stuffz
Thalf2 = np.log(0.5)*c/np.real(eigenvalue2)/V0

Period1 = []
Period2 = []

for i in xrange(len(eigenvalue1)):
    if np.imag(eigenvalue1[i])!=0.:
        Period1.append(2.*np.pi*c/np.imag(eigenvalue1[i])/c/V0) #period stuffz
    else:
        Period1.append(0)
for i in xrange(len(eigenvalue1)):
    if np.imag(eigenvalue2[i])!=0.:
        Period2.append(2.*np.pi*c/np.imag(eigenvalue2[i])/c/V0)
    else:
        Period2.append(0)

# responses, inputs are actual deflections
u1 = list(de)
u2 = np.vstack((da,dr))

x01 = np.matrix([[0],
                 [0],
                 [0],
                 [q0*c/V0]])

x02 = np.matrix([[0],
                 [0],
                 [p0*b/2./V0],
                 [r0*b/2./V0]])

y1,t1,x0 = cm.lsim(SYM,u1,t,x01)
y2,t1,x0 = cm.lsim(ASYM,u2.T,t,x02)
```

```
    #adjusting for actual values

    y1[:,0] = (y1[:,0]*V0)+V0   #u to velocity
    y1[:,1] = y1[:,1]+alpha0    #aoa
    y1[:,2] = y1[:,2]+th0       #pitch angle
    y1[:,3] = y1[:,3]*V0/c        #pitch rate
    y2[:,0] = y2[:,0]           #sideslip (assumed 0 initially)
    y2[:,1] = y2[:,1]+phi0      #roll angle
    y2[:,2] = y2[:,2]*2*V0/b        #roll rate
    y2[:,3] = y2[:,3]*2*V0/b        #yaw rate

    return(y1,y2,t,eigenvalue1,eigenvalue2,V0)
```

Main Simulation Loop: 'loop.py'

```
# Citation 550 - Linear simulation
import numpy as np
import control.matlab as cm
import scipy.linalg as la
from matplotlib import pyplot as plt
import flightdata
from simulator import simulate as sim

#importing data from the flight test for input and comparison

data    = flightdata.inp()
t       = data[0] #time (s)


#all the important testdata
h  = data[1] #pressure altitude  (m)
T  = data[2] #temperature (K)
V  = data[3] #velocity (m/s)
de = data[4] #elevator deflection (rad)
da = data[5] #aileron deflection (rad)
dr = data[6] #rudder deflection (rad)
p  = data[7] #roll rate (rad/s)
q  = data[8] #pitch rate (rad/s)
r  = data[9] #yaw rate (rad/s)
theta = data[10] #pitch angle (rad)
phi   = data[11] #roll angle (rad)
alpha = data[12] #angle of attack (rad)
fused = data[13] #fuel used (kg)

###plot full data set, chosen data: elevator input, velocity output
##plt.subplot(211)
##plt.plot(t,de)
##plt.ylabel('rad')
##plt.title("Elevator deflection")
##plt.subplot(212)
##plt.plot(t,V)
##plt.xlabel('t')
##plt.ylabel('m/s')
##plt.title("Velocity")
##plt.show()
```

```
#all manoevres, identifying indexes in lists:

dutch = []
for i in xrange(len(t)):
    if t[i]>2622. and t[i]<2647.:
        dutch.append(i)

aperiodic = []
for i in xrange(len(t)):
    if t[i]>2991. and t[i]<3007.:
        aperiodic.append(i)

spiral = []
for i in xrange(len(t)):
    if t[i]>3070. and t[i]<3200.:
        spiral.append(i)

short = []
for i in xrange(len(t)):
    if t[i]>2556. and t[i]<2567.:
        short.append(i)

phugoid = []
for i in xrange(len(t)):
    if t[i]>2763. and t[i]<2900.:
        phugoid.append(i)

# print + plot system eigenvalues
sym = np.array(sim(short)[3]) #Random manoevre picked, since values should not change much
asym = np.array(sim(short)[4]) #only decrease in fuel matters
print "Symmetric eigenvalues: ",sym
print "Assymetric eigenvalues: ",asym

plt.title("Symmetric System Eigenvalues")
plt.plot(np.real(sym),np.imag(sym),'o')
plt.plot([-5,5],[0,0])
plt.plot([0,0],[-3,3])
plt.xlabel("Real")
plt.ylabel("Imaginary")
plt.show()
plt.title("Asymmetric System Eigenvalues")
plt.plot(np.real(asym),np.imag(asym),'o')
plt.plot([-5,5],[0,0])
plt.plot([0,0],[-3,3])
plt.xlabel("Real")
plt.ylabel("Imaginary")
plt.show()


#plot per manoevre:

#phugoid
plt.suptitle('No disturbance after steady straight flight')
plt.subplot(231)
plt.plot(t[phugoid[0]:phugoid[-1]]-t[phugoid[0]],V[phugoid[0]:phugoid[-1]]\
,label='Testdata')
```

```
plt.plot(sim(phugoid)[2]-t[phugoid[0]],sim(phugoid)[0][:,0]\
,label='Simulated')
plt.xlabel('s')
plt.ylabel('m/s')
plt.title("Velocity")
plt.legend(loc=4,prop={'size':12})
plt.subplot(232)
plt.plot(t[phugoid[0]:phugoid[-1]]-t[phugoid[0]],alpha[phugoid[0]:phugoid[-1]]\
,label='Testdata')
plt.plot(sim(phugoid)[2]-t[phugoid[0]],sim(phugoid)[0][:,1],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Angle of attack")
plt.legend(loc=4,prop={'size':12})
plt.subplot(233)
plt.plot(t[phugoid[0]:phugoid[-1]]-t[phugoid[0]],theta[phugoid[0]:phugoid[-1]],label='Testdata')
plt.plot(sim(phugoid)[2]-t[phugoid[0]],sim(phugoid)[0][:,2],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Pitch angle")
plt.legend(loc=4,prop={'size':12})
plt.subplot(234)
plt.plot(t[phugoid[0]:phugoid[-1]]-t[phugoid[0]],q[phugoid[0]:phugoid[-1]],label='Testdata')
plt.plot(sim(phugoid)[2]-t[phugoid[0]],sim(phugoid)[0][:,3],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("pitch rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(235)
plt.plot(t[phugoid[0]:phugoid[-1]]-t[phugoid[0]],de[phugoid[0]:phugoid[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("de")
plt.show()

#short period
plt.suptitle('Short Period')
plt.subplot(231)
plt.plot(t[short[0]:short[-1]]-t[short[0]],V[short[0]:short[-1]],label='Testdata')
plt.plot(sim(short)[2]-t[short[0]],sim(short)[0][:,0],label='Simulated')
plt.xlabel('s')
plt.ylabel('m/s')
plt.title("Velocity")
plt.legend(loc=3,prop={'size':12})
plt.subplot(232)
plt.plot(t[short[0]:short[-1]]-t[short[0]],alpha[short[0]:short[-1]],label='Testdata')
plt.plot(sim(short)[2]-t[short[0]],sim(short)[0][:,1],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Angle of attack")
plt.legend(loc=4,prop={'size':12})
plt.subplot(233)
plt.plot(t[short[0]:short[-1]]-t[short[0]],theta[short[0]:short[-1]],label='Testdata')
plt.plot(sim(short)[2]-t[short[0]],sim(short)[0][:,2],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
```

```
plt.title("Pitch angle")
plt.legend(loc=4,prop={'size':12})
plt.subplot(234)
plt.plot(t[short[0]:short[-1]]-t[short[0]],q[short[0]:short[-1]],label='Testdata')
plt.plot(sim(short)[2]-t[short[0]],sim(short)[0][:,3],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("pitch rate")
plt.subplot(235)
plt.plot(t[short[0]:short[-1]]-t[short[0]],de[short[0]:short[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("de")
plt.show()

#Aperiodic roll
plt.suptitle('No disturbance after steady straight flight')
plt.subplot(231)
plt.plot(t[aperiodic[0]:aperiodic[-1]]-t[aperiodic[0]],phi[aperiodic[0]:aperiodic[-1]]\
,label='Testdata')
plt.plot(sim(aperiodic)[2]-t[aperiodic[0]],sim(aperiodic)[1][:,1]\
,label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Roll angle")
plt.legend(loc=4,prop={'size':12})
plt.subplot(232)
plt.plot(t[aperiodic[0]:aperiodic[-1]]-t[aperiodic[0]],p[aperiodic[0]:aperiodic[-1]]\
,label='Testdata')
plt.plot(sim(aperiodic)[2]-t[aperiodic[0]],sim(aperiodic)[1][:,2],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Roll rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(233)
plt.plot(t[aperiodic[0]:aperiodic[-1]]-t[aperiodic[0]],r[aperiodic[0]:aperiodic[-1]]\
,label='Testdata')
plt.plot(sim(aperiodic)[2]-t[aperiodic[0]],sim(aperiodic)[1][:,3],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Yaw rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(234)
plt.plot(t[aperiodic[0]:aperiodic[-1]]-t[aperiodic[0]],da[aperiodic[0]:aperiodic[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("da")
plt.subplot(235)
plt.plot(t[aperiodic[0]:aperiodic[-1]]-t[aperiodic[0]],dr[aperiodic[0]:aperiodic[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("dr")
plt.show()
##
#Dutch roll
plt.suptitle('Dutch Roll')
```

```
plt.subplot(231)
plt.plot(t[dutch[0]:dutch[-1]]-t[dutch[0]],phi[dutch[0]:dutch[-1]],label='Testdata')
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,1],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Roll angle")
plt.legend(loc=4,prop={'size':12})
plt.subplot(232)
plt.plot(t[dutch[0]:dutch[-1]]-t[dutch[0]],p[dutch[0]:dutch[-1]],label='Testdata')
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,2],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Roll rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(233)
plt.plot(t[dutch[0]:dutch[-1]]-t[dutch[0]],r[dutch[0]:dutch[-1]],label='Testdata')
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,3],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Yaw rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(234)
plt.plot(t[dutch[0]:dutch[-1]]-t[dutch[0]],da[dutch[0]:dutch[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("da")
plt.subplot(235)
plt.plot(t[dutch[0]:dutch[-1]]-t[dutch[0]],dr[dutch[0]:dutch[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("dr")
plt.show()

#spiral
plt.suptitle('Spiral')
plt.subplot(231)
plt.plot(t[spiral[0]:spiral[-1]]-t[spiral[0]],phi[spiral[0]:spiral[-1]],label='Testdata')
plt.plot(sim(spiral)[2]-t[spiral[0]],sim(spiral)[1][:,1],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("Roll angle")
plt.legend(loc=3,prop={'size':12})
plt.subplot(232)
plt.plot(t[spiral[0]:spiral[-1]]-t[spiral[0]],p[spiral[0]:spiral[-1]],label='Testdata')
plt.plot(sim(spiral)[2]-t[spiral[0]],sim(spiral)[1][:,2],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Roll rate")
plt.legend(loc=4,prop={'size':12})
plt.subplot(233)
plt.plot(t[spiral[0]:spiral[-1]]-t[spiral[0]],r[spiral[0]:spiral[-1]],label='Testdata')
plt.plot(sim(spiral)[2]-t[spiral[0]],sim(spiral)[1][:,3],label='Simulated')
plt.xlabel('s')
plt.ylabel('Rad/s')
plt.title("Yaw rate")
plt.legend(loc=3,prop={'size':12})
```

```
plt.subplot(234)
plt.plot(t[spiral[0]:spiral[-1]]-t[spiral[0]],da[spiral[0]:spiral[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("da")
plt.subplot(235)
plt.plot(t[spiral[0]:spiral[-1]]-t[spiral[0]],dr[spiral[0]:spiral[-1]])
plt.xlabel('s')
plt.ylabel('Rad')
plt.title("dr")
plt.show()
```

Disturbance Response Simulation: 'loop-disturbance input.py'

```
#BEFORE RUNNING READ ME
#for the disturbance inputs, the rudder and aileron deflection have to be manually
#set to zero for the whole time period in the 'simulation.py' code. Besides this
#the X01 value representing alpha or angle of attack can be changed from zero to
#any initial angle of attack to represent disturbances.

import numpy as np
import control.matlab as cm
import scipy.linalg as la
from matplotlib import pyplot as plt
import flightdata
from simulator import simulate as sim

#importing data from the flight test for input and comparison

data    = flightdata.inp()
t       = data[0] #time (s)


#all the important testdata
h  = data[1] #pressure altitude  (m)
T  = data[2] #temperature (K)
V  = data[3] #velocity (m/s)
de = data[4] #elevator deflection (rad)
da = data[5] #aileron deflection (rad)
dr = data[6] #rudder deflection (rad)
p  = data[7] #roll rate (rad/s)
q  = data[8] #pitch rate (rad/s)
r  = data[9] #yaw rate (rad/s)
theta = data[10] #pitch angle (rad)
phi   = data[11] #roll angle (rad)
alpha = data[12] #angle of attack (rad)
fused = data[13] #fuel used (kg)

#identifying random time interval for disturbance inputs, ignore the name 'dutch'

dutch = []
for i in xrange(len(t)):
    if t[i]>3300. and t[i]<3400.:
        dutch.append(i)

#plot for symmetric and asymmetric:
```

```
#symmetric
plt.suptitle('No disturbance after steady straight flight')
plt.subplot(231)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[0][:,0])
#plt.xlabel('s')
#plt.ylabel('m/s')
plt.title("Velocity")
plt.subplot(232)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[0][:,1],)
#plt.xlabel('s')
#plt.ylabel('Rad')
plt.title("Angle of attack")
plt.subplot(233)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[0][:,2])
#plt.xlabel('s')
#plt.ylabel('Rad')
plt.title("Pitch angle")
plt.subplot(234)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[0][:,3])
#plt.xlabel('s')
#plt.ylabel('Rad/s')
plt.title("pitch rate")
plt.subplot(235)
plt.plot(t[dutch[0]:dutch[-1]+1]-t[dutch[0]],de[0]*np.ones(len(dutch)))
#plt.xlabel('s')
#plt.ylabel('Rad')
plt.title("de")
plt.show()


#Assymetric
plt.suptitle('No disturbance after steady straight flight')
plt.subplot(231)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,1])
#plt.xlabel('s')
#plt.ylabel('Rad')
plt.title("Roll angle")
plt.subplot(232)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,2])
#plt.xlabel('s')
#plt.ylabel('Rad/s')
plt.title("Roll rate")
plt.subplot(233)
plt.plot(sim(dutch)[2]-t[dutch[0]],sim(dutch)[1][:,3])
#plt.xlabel('s')
#plt.ylabel('Rad/s')
plt.title("Yaw rate")
plt.subplot(234)
plt.plot(t[dutch[0]:dutch[-1]+1]-t[dutch[0]],0*np.ones(len(dutch)))
#plt.xlabel('s')
#plt.ylabel('Rad')
plt.title("da")
plt.subplot(235)
plt.plot(t[dutch[0]:dutch[-1]+1]-t[dutch[0]],0*np.ones(len(dutch)))
#plt.xlabel('s')
```

```
#plt.ylabel('Rad')
plt.title("dr")
plt.show()
```

Eigenvalue Comparison: 'eigenvalue determination.py'

```
from simulator import simulate
from analytical import analytical
import numpy as np
import control.matlab as cm
import scipy.linalg as la
from matplotlib import pyplot as plt
import flightdata

b=15.911
#the time periods of each maneuver
#SP, PH, AR, DR, AS
epsilon = [[25500,25501,25502],[27531, 27532, 27533],[29900, 29901,29902]\
,[26300, 26301,26302], [31100, 31101, 31102]]
print "------------NUMERICAL-----------------------------"
for i in epsilon:
    if i == epsilon[0]: #short period sym
        real = simulate(i)[3][0].real
        imag = simulate(i)[3][0].imag

        Tt = np.log(0.5)/(real)
        Pp = 2*np.pi/(imag)
        damp = -real/(np.sqrt(real**2+imag**2))
        omega_o = np.sqrt(real**2+imag**2)
        omega_n = omega_o*np.sqrt(1-damp**2)
        Cc = float(Tt/Pp)
        print "Short Period : ", simulate(i)[3][0]
        print "T 0.5 : ", Tt
        print "Period : ", Pp
        #print "delta : ",analytical(i)[4]
        print "damp : ", damp
        print "omega_n : ", omega_n
        #print "Cc : ",analytical(i)[7]
        print
    if i == epsilon[1]: #phugoid sym
        real = simulate(i)[3][2].real
        imag = simulate(i)[3][2].imag

        Tt = np.log(0.5)/(real)
        Pp = 2*np.pi/(imag)
        damp = -real/(np.sqrt(real**2+imag**2))
        omega_o = np.sqrt(real**2+imag**2)
        omega_n = omega_o*np.sqrt(1-damp**2)
        Cc = float(Tt/Pp)
        print "Phugoid : ", simulate(i)[3][2]
        print "T 0.5 : ", Tt
        print "Period : ", Pp
        #print "delta : ",analytical(i)[4]
        print "damp : ", damp
        print "omega_n : ", omega_n
        #print "Cc : ",analytical(i)[7]
        print
```

```
    if i == epsilon[2]:

        Tt = np.log(0.5)/(simulate(i)[4][0])
        taut = -b/(simulate(i)[4][0]*simulate(i)[-1])
        print "Aperiodic Roll : ", simulate(i)[4][0]
        print "T 0.5 : ", Tt
        print "Tau: : ", taut
        print

    if i == epsilon[3]:
        real = simulate(i)[4][1].real
        imag = simulate(i)[4][1].imag

        Tt = np.log(0.5)/(real)
        Pp = 2*np.pi/(imag)
        damp = -real/(np.sqrt(real**2+imag**2))
        omega_o = np.sqrt(real**2+imag**2)
        omega_n = omega_o*np.sqrt(1-damp**2)
        Cc = float(Tt/Pp)
        print "Dutch Roll : ", simulate(i)[4][1]
        print "T 0.5 : ", Tt
        print "Period : ", Pp
        #print "delta : ",analytical(i)[4]
        print "damp : ", damp
        print "omega_n : ", omega_n
        #print "Cc : ",analytical(i)[7]
        print

    if i == epsilon[4]:

        Tt = np.log(0.5)/(simulate(i)[4][3])
        taut = -b/(simulate(i)[4][3]*simulate(i)[-1])
        print "Aperiodic Spiral : ", simulate(i)[4][3]
        print "T 0.5 : ", Tt
        print "Tau: : ", taut
        print
print "----------ANALYTICAL----------"
for i in epsilon:
    if i == epsilon[0] or i== epsilon[1] or i == epsilon[3]:
        print analytical(i)[0]
        print analytical(i)[1]
        print "T 0.5 : ", analytical(i)[2]
        print "Period : ",analytical(i)[3]
        #print "delta : ",analytical(i)[4]
        print "damp : ",analytical(i)[5]
        print "omega_n : ",analytical(i)[6]
        #print "Cc : ",analytical(i)[7]
        print
    if i == epsilon[2] or i == epsilon[4]:
        print analytical(i)[0]
        print analytical(i)[1]
        print "T 0.5 : ", analytical(i)[2]
        print "Tau : ", analytical(i)[8]
```

Aerodynamic Coefficient Calculation: 'measurement_series_1_FINAL.py'

```
import numpy as np
from math import *
import matplotlib.pyplot as plt
import matplotlib

S      = 30. #[m2]
c      = 2.0569 #[m]
b      = 15.911 #[m]
g      = 9.80665    #[m/s2]
A      = b**2/S #[-]

lam    = -6.5/1000.
T0     = 288.15 #[K]
p0     = 101325. #[Pa]
rho0   = 1.225 #[kg/m3]
g      = 9.80665 #[m/s2]
R      = 287.05 #[J/kgK]
gamma  = 1.4

Ws = 60500 #[N]

BEW = 9165.00*0.453592 #[kg] CHANGE!!
WFuel = 2700.00*0.453592 #[kg] CHANGE!!
WP1 = 99#[kg]
WP2 = 90#[kg]
WC = 99#[kg]
WO1L = 74#[kg]
WO1R = 66#[kg]
WO2L = 79#[kg]
WO2R = 80#[kg]
WO3L = 91#[kg]
WO3R = 84#kg]

Wtotal = BEW+WFuel+WP1+WP2+WC+WO1L+WO1R+WO2L+WO2R+WO3L+WO3R #[kg]


#pressure altitudes
hpft = [5010.,5010.,5010.,5000.,5010.,5050.] #[ft]
hp = 0.3048*np.array(hpft) #[m]

#Caliberated airspeed
Vckts = [251,225,191,160,137,114] #[kts]
Vc = 0.514444*np.array(Vckts) #[m/s]

#Total air temperature
TATdeg = [11.0,10.2,8.5,6.8,6.0,4.2] #[C]
TAT = 273+np.array(TATdeg) #[K]

#Fuel used
Fupounds = [317,349,376,397,418,440] #[lbs]
Fu =  0.453592*np.array(Fupounds)#[kg]

#Fuel flow left
Fflpph = [755,650,521,449,423,419] #[lbs/hr]
Ffl = (0.453592/3600.)*np.array(Fflpph) #[kg/s]
```

```
#Fuel flow right
Ffrpph = [819,711,567,504,478,486] #[lbs/hr]
Ffr = (0.453592/3600.)*np.array(Ffrpph) #[kg/s]

#angles of attack
aoa = np.array([0.8,1.4,2.6,4.4,6.4,10.0]) #[degrees]

#ISA temperature
Tisa = T0 + lam*hp #[K]

#pressure
p = p0*(1+(lam*hp)/(T0))**(-g/(lam*R)) #[Pa]

#Mach number
M = np.sqrt(2./(gamma-1.)*((1.+p0/p*((1.+((gamma-1)/(2.*gamma))\
*rho0/p0*Vc**2)**(gamma/(gamma-1.))-1.))**((gamma-1.)/gamma)-1.))

#Static temperature
T = TAT/(1.+(gamma-1.)/2.*M**2) #[K]

#Density
rho = p/R/T #[kg/m3]

#Speed of sound
a = np.sqrt(gamma*R*T) #[m/s]

#True airspeed
Vt = M*a #[m/s]

#Equivalent Airspeed
Ve = Vt*np.sqrt(rho/rho0) #[m/s]

#Reynolds number
mu = 1.458*10**(-6)*np.sqrt(T**3)/(T+110.4) #change it
Re = (rho*Vt*c)/mu

#Instantaneous Weight
Wi = Wtotal - Fu

#Thrust from (thrust.exe)
Thl = np.array([3372.87,2817.55,2125.56,1779.08,1706.25,1805.96]) #[N]
Thr = np.array([3815.37,3235.2,2450.28,2179.5,2120.92,2333.85]) #[N]

#Lift coefficient
CL = (Wi*g)/(0.5*rho*Vt**2*S)
CL2 = CL**2

#Drag coefficient
CDi = (Thl+Thr)/(0.5*rho*Vt**2*S)

######################### PLOTS #############################
matplotlib.rc('xtick', labelsize=15)
matplotlib.rc('ytick', labelsize=15)

CD0theoretical = 0.04
Clalphatheoretical = 5.084
```

```
etheoretical = 0.8
aoa0theoretical = -1.3

#CL-alpha plot
CLalpha = np.polyfit(aoa, CL, 1)
Lslope  = CLalpha[0]
Lslopeinrad = CLalpha[0]*(180/pi)
aoa0    = -CLalpha[1]/CLalpha[0]

u = np.arange(0,10.5,0.5)
v = Lslope*(u-aoa0)
vtheoretical = Clalphatheoretical*(pi/180)*(u-aoa0theoretical)



plt.subplot(221)
plt.plot(u, v,'r',linewidth=2.0,label='Experimental trendline')
plt.plot(u, vtheoretical,'b--',linewidth=2.0,label='Theoretical trendline')
plt.scatter(aoa,CL,s=50,color='green',label = 'Measurement points')
plt.legend(loc = 'upper left')
plt.ylabel("CL [-]",fontsize=20)
plt.xlabel("Alpha [deg]",fontsize=20)

#CD-CL^2 plot
CL2plot = np.polyfit(CL2, CDi, 1)
e       = 1./(np.pi*A*CL2plot[0])
CD0     = CL2plot[1]

x = np.arange(-0.2,1.05,0.05)
y = CL2plot[0]*x + CD0

ytheoretical = CD0theoretical + x*(1./pi/A/etheoretical)

plt.subplot(222)
plt.plot(x,y,'r',linewidth=2.0,label='Experimental trendline')
plt.plot(x, ytheoretical,'b--',linewidth=2.0,label='Theoretical trendline')
plt.scatter(CL2,CDi,s=50,color='green',label = 'Measurement points')
plt.legend(loc = 'upper left')
plt.xlabel("CL^2 [-]",fontsize=20)
plt.ylabel("CD [-]",fontsize=20)

#Draf coeff again
CD = CD0 + CL**2/(pi*A*e)

CDalpha = np.polyfit(aoa,CD,2)

l = np.arange(0,10.5,0.5)
m = CDalpha[0]*l**2 + CDalpha[1]*l + CDalpha[2]
mtheoretical = CD0theoretical + (1./pi/A/etheoretical)*(Clalphatheoretical*(pi/180)*(l-aoa0))**2

#CD-alpha plot
plt.subplot(223)
plt.plot(l, m,'r',linewidth=2.0,label='Experimental curvefit')
plt.plot(l, mtheoretical,'b--',linewidth=2.0,label='Theoretical curvefit')
plt.scatter(aoa,CDi,s=50,color='green',label = 'Measurement points')
plt.legend(loc = 'upper left')
```

```
plt.xlabel("Alpha [deg]",fontsize=20)
plt.ylabel("CD [-]",fontsize=20)

#CL-CD plot
CLCD = np.polyfit(CL,CD,2)

s = np.arange(0.1,1.0,0.05)
t = CLCD[0]*s**2 + CLCD[2]
ttheoretical = CD0theoretical+s**2*(1./pi/A/etheoretical)

plt.subplot(224)
plt.plot(t, s,'r',linewidth=2.0,label='Experimental curvefit')
plt.plot(ttheoretical, s,'b--',linewidth=2.0,label='Theoretical curve fit')
plt.scatter(CDi,CL,s=50,color='green',label = 'Measurement points')
plt.legend(loc = 'upper left')
plt.xlabel("CD [-]",fontsize=20)
plt.ylabel("CL [-]",fontsize=20)
plt.show()
```

Aerodynamic Coefficient Calculation 2: 'measurement_series_2_FINAL.py'

```
import numpy as np
from math import *
import scipy.optimize as optimization
import matplotlib
import matplotlib.pyplot as plt

S      = 30. #[m2]
c      = 2.0569 #[m]
b      = 15.911 #[m]
g      = 9.80665    #[m/s2]
A      = b**2/S #[-]

lam     = -6.5/1000.
T0    = 288.15 #[K]
p0    = 101325. #[Pa]
rho0 = 1.225 #[kg/m3]
g     = 9.81 #[m/s2]
R     = 287.05 #[J/kgK]
gamma = 1.4

Ws = 60500 #[N]

BEW = 9165.00*0.453592 #[kg] CHANGE!!
WFuel = 2700.00*0.453592 #[kg] CHANGE!!
WP1 = 99#[kg]
WP2 = 90#[kg]
WC = 99#[kg]
WO1L = 74#[kg]
WO1R = 66#[kg]
WO2L = 79#[kg]
WO2R = 80#[kg]
WO3L = 91#[kg]
WO3R = 84#kg]

Wtotal = BEW+WFuel+WP1+WP2+WC+WO1L+WO1R+WO2L+WO2R+WO3L+WO3R #[kg]
```

```
#pressure altitudes
hpft = [7040.,7300.,7630.,7980.,7170.,6670.,6200.,6720.,6820.] #[ft]
hp = 0.3048*np.array(hpft) #[m]

#Caliberated airspeed
Vckts = [170.,160.,150.,141.,183.,190.,198.,170.,170.] #[kts]
Vc = 0.514444*np.array(Vckts) #[m/s]

#Total air temperature
TATdeg = [4.5,3.5,2.0,1.2,4.8,6.0,7.5,4.5,4.8] #[C]
TAT = 273+np.array(TATdeg) #[K]

#Fuel used
Fupounds = [537.,552.,570.,583.,612.,647.,663.,687.,715.] #[lbs]
Fu =  0.453592*np.array(Fupounds)#[kg]

#Fuel flow left
Fflpph = [443.,439.,434.,429.,446.,455.,462.,450.,449.] #[lbs/hr]
Ffl = (0.453592/3600)*np.array(Fflpph) #[kg/s]

#Fuel flow right
Ffrpph = [492.,487.,483.,478.,494.,503.,508.,497.,494.]
Ffr = (0.453592/3600)*np.array(Ffrpph) #[kg/s]

#angles of attack
aoa = np.array([3.6,4.4,5.2,5.9,2.9,2.5,2.2,3.6,3.6])

#ISA temperature
Tisa = T0 + lam*hp #[K]

#pressure
p = p0*(1+(lam*hp)/(T0))**(-g/(lam*R)) #[Pa]

#Mach number
M = np.sqrt(2./(gamma-1.)*((1.+p0/p*((1.+((gamma-1)/\
(2.*gamma))*rho0/p0*Vc**2)**(gamma/(gamma-1.))-1.))**((gamma-1.)/gamma)-1.))

#Static temperature
T = TAT/(1.+(gamma-1.)/2.*M**2) #[K]

#Density
rho = p/R/T #[kg/m3]

#Speed of sound
a = np.sqrt(gamma*R*T) #[m/s]

#True airspeed
Vt = M*a #[m/s]

#Equivalent Airspeed
Ve = Vt*np.sqrt(rho/rho0) #[m/s]

#Reynolds number
mu = 1.458*10**(-6)*np.sqrt(T**3)/(T+110.4) #change it
Re = (rho*Vt*c)/mu
```

```
#Instantaneous Weight
Wi = Wtotal - Fu

#Thrust from (thrust.exe)
Thl = np.array([1780.43,1815.61,1849.77,1877.34,1743.37,1748.59,1734.18,1818.14,1813.2]) #[N]
Thr = np.array([2133.51,2166.74,2215.44,2246.73,2082.76,2083.42,2051.17,2156.67,2136.69]) #[N]

#Standard thrust from (thrust.exe)
Thls = np.array([1343.22, 1399.46, 1462.01, 1518.63, 1294.78, 1244.73, 1191.64, \
1331.73, 1333.89]) #[N]
Thrs = np.array([1343.22, 1399.46, 1462.01, 1518.63, 1294.78, 1244.73, 1191.64,\
1331.73, 1333.89]) #[N]

#Lift coefficient
CL = (Wi*g)/(0.5*rho*Vt**2*S)
CL2 = CL**2

#Drag coefficient
CDi = (Thl+Thr)/(0.5*rho*Vt**2*S)

#Reduced aircraft mass implented to equivalent airspeed
Vetilde = Ve*np.sqrt(Ws/(Wtotal*g))

#Elevator deflection from measurements
deltae = np.array([-0.2,-0.5,-0.9,-1.3,0.2,0.4,0.6,-0.1,-0.6]) #[deg]

Cmtc = -0.0064 #[-]
D = 0.686 #[m]
Tcs  = (Thls+Thrs)/(rho0*Vetilde**2*D**2) #[-]
Tc   = (Thl+Thr)/(rho*Vt**2*D**2) #[-]

#Change in elevator deflection before and after the shift in c.g.
ddeltae = -0.6 - (-0.1)#[deg]

#Change in a/c centre of gravity
dxcg = 0.4267 - 0.4892 #[m]

#Elevator effectiveness
CN = (CL[-1]+CL[-2])/2

Cmdelta = -(1./(ddeltae*pi/180))*CN*(dxcg/c)

#Longitudinal stability
#Cmalpha =

#Reduced elevator deflection
deltaestar = deltae - (1/Cmdelta)*(Cmtc*(Tcs - Tc))

#Control force measurements
Fe = np.array([0,-22,-34,-42,32,52,70,0,-27]) #[N]

#Reduced control force
Festar = Fe*(Ws/(Wtotal*g))

mdeaoa = np.polyfit(aoa[:7],deltae[:7],1)
```

```
Cmalpha = -mdeaoa[0]*Cmdelta

def func(V,a,b):
    return a*(1./V**2) + b

test = optimization.curve_fit(func,Vetilde,deltaestar,[0,0])

l = -1.99892979e+04 #from test
b = 2.42829159 #from test

x= np.arange(70,110,1)
y= l*(1./x**2) + b


matplotlib.rc('xtick', labelsize=15)
matplotlib.rc('ytick', labelsize=15)


#deltae*-Vetilde plot

plt.subplot(121)
plt.gca().invert_yaxis()
plt.plot(x,y,'r',linewidth=2.0,label='Experimental curvefit')
plt.scatter(Vetilde[:7],deltaestar[:7],s=50 ,color = "green",label = 'Measurement points')
plt.legend(loc = 'upper left')
plt.xlabel("Reduced equivalent airspeed [m/s]",fontsize=20)
plt.ylabel("Reduced elevator deflection [deg]",fontsize=20)

def func(V,p,q):
    return p*V**2 + q

test2 = optimization.curve_fit(func,Vetilde,Festar,[0,0])

m = 2.32693261e-02 #from test2
q = -1.76843514e+02#from test2

u= np.arange(70,110,1)
v= m*x**2 + q

#Fe-Vetilde plot
plt.subplot(122)
plt.gca().invert_yaxis()
plt.plot(u,v,'r',linewidth=2.0,label='Experimental curvefit')
plt.scatter(Vetilde[:7], Festar[:7], s=50 ,color = "green",label = 'Measurement points')
plt.legend(loc = 'upper left')
plt.xlabel("Reduced equivalent airspeed [m/s]",fontsize=20)
plt.ylabel("Reduced control Force, Fe [N]",fontsize=20)

plt.show()
```

1003_F2

GRADEMARK REPORT

FINAL GRADE

91 /100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

Comment 1 | Rep. overview

out of page margins!

Comment 2 | Rep. overview

All the inuts have been defined very clearly.

PAGE 5

Comment 3 | Rep. overview

There are other outputs when considerins the first and second stationary measurements

Comment 4 | Analyt. mod.

The aerodynamics references frame is also one that should have been used in this project as both lift and drag are determined in this reference frame.

PAGE 6

Comment 5

are

Comment 6 | Analyt. mod.

Very extended list of assumptions with a lot of explanation for all of them.

Comment 7 | Analyt. mod.

contradictiory assumptions: flat and spherical Earth. Use flat only!

PAGE 7

Comment 8 | Analyt. mod.

this is the actual assumption, the other is a consequence from it.

PAGE 8

Comment 9

Kinetic diagran was not necessary, but it is a good bonus

PAGE 9

Comment 10

wrong name

PAGE 10

Comment 11 | Analyt. mod.

Good idea

PAGE 11

Comment 12 | Rep. overview

page margins!

PAGE 12

PAGE 13

Comment 13 | Analyt. mod.

It is the other way around. Experimental gives lower drag values.

Comment 14 | Analyt. mod.

There are no flowcharts explaining the process to obtain the curves neither for the first nor the second set of stationary measurements. No explanation of the difference between experimental and theoretical for the second set of measurements.

PAGE 14

Comment 15 | Missing data

Only three different groups of data are evaluated

PAGE 15

PAGE 16

Comment 16 | Analyt. mod.

Very nice explanation, but the final expresion for the eigenvalues is missing

Comment 17 | Analyt. mod.

Very nice explanation, but the final expresion for the eigenvalues is missin. This is the case for the asymmetric modes also, except for the spiral.

PAGE 17

PAGE 18

Comment 18

Please be a little bit more technical.

Comment 19 | Analyt. mod.

How is this time constant defined?

Comment 20 | Analyt. mod.

effects of assumptions was a separate section in last year's reports, now they are required to be explained together with the assumptions.

Comment 21 | Num. mod.

assumptions for numerical model? liniarization of state space representation as LTI should be stated with effects

PAGE 19

PAGE 20

Comment 22 | Num. mod.

Very useful way to plot the eigenvalues

PAGE 21

Comment 23 | Num. mod.

the numerical part is very good but because of the computational model it will score excellent.

PAGE 22

Comment 24 | Validation

Very important to do this in order to validate the model

Comment 25

two or four?

PAGE 23

Comment 26 | Num. mod.

Incredible flowchart, it seems that inside this one all the processes are defined instead of having three different ones.

PAGE 24

Comment 27 | Verification

Great number of unit tests. There are three system tests which is how it is supposed to be

PAGE 25

Comment 28 | Verification

There is always bigger percentage difference for values that are very close to zero.

Comment 29 | Validation

From half an oscillation information coul dhave been extracted

Comment 30

j

PAGE 26

Comment 31 | Validation

If you think about the intrument used to compute the angle of attack, you will realized that is not the most accurate method and that as it is next to the fuselage maybe there was some interaction relating also boundary layer effects in the process.

Comment 32 | Rep. overview

please show the graphs in degrees next time

Comment 33 | Validation

high frequency

PAGE 27

Comment 34

oscillations can be due to a slight Dutch Roll motion starting on a first stage

PAGE 28

PAGE 29

Comment 35 | Validation

0.251 rad is approx. 14 degrees, this is quite a large disturbance and specially a linear model would not give reasonable results. Bear in mind that the stall AoA for the Cessna is around 12 degrees

Comment 36 | Validation

I do not understand what the undisturbed response represents

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

Comment 37 | Validation

Very interesting recomendations

PAGE 35

Comment 38

omg, Thomas! it's spring outside, enjoy life! ;) but on a more serious note, amazing contribution! Congrats

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

RUBRIC: AE3212-II FINAL REPORT 2017_FD 90.50 / 100

REP. OVERVIEW (5%) 80 / 100

Overview of the entire report

MISSING (0)	Missing task division, not everyone assigned to at least two packages out of simulation, verification and validation and/or going over the page limit
INSUFFICIENT (40)	Major parts of the report are missing. Formatting of the report very poor/missing Many spelling and grammatical errors Task division present, but must be improved
SUFFICIENT (60)	Meets basic requirements of report Structure sufficient Quite some mistakes and/or spelling and grammatical errors Task division present, but needs some revision
MORE THAN SUFFICIENT (70)	Report exceeds basic requirements Good structure Some mistakes and/or spelling and grammatical errors Clear task division present, but can be improved
GOOD (80)	Good report Good structure and layout Few mistakes and/or minor spelling and grammatical errors Clear task division present
VERY GOOD (90)	Excellent report Very good structure and layout No spelling and grammatical errors Very clear task division present
EXCELLENT (100)	Exemplary report Excellent structure and layout No spelling and grammatical errors Exemplary task division present

ANALYT. MOD. (25%) 80 / 100

Analytical model

MISSING (0)	Assumptions analytical approach missing and/or theory behind analytical approach missing and/or analytical results missing and/or effects of assumptions on analytical approach missing
INSUFFICIENT (40)	Insufficient number of assumptions and/or inconsistent assumptions and/or mistakes in the assumptions made Explanation of theory incomplete or with considerable mistakes Analytical results incomplete or with considerable mistakes Explanation of some of the effects of assumptions on analytical approach missing or with considerable mistakes
SUFFICIENT (60)	Reasonable number of assumptions, but some mistakes in the assumptions made Explanation of theory sufficient, but with mistakes Analytical results sufficient, but with some mistakes Explanation of effects of assumptions on analytical approach sufficient, but with mistakes
MORE THAN SUFFICIENT (70)	Reasonable number of assumptions and more than sufficient motivation Explanation of theory more than sufficient, but with room for improvement, some mistakes made Analytical results more than sufficient, but with room for improvement, some mistakes made Explanation of effects of assumptions on analytical approach more than sufficient, but with room for improvement, some mistakes made
GOOD (80)	Reasonable number of assumptions and good motivation Explanation of theory good, minor mistakes made Analytical results good, minor mistakes made

	Explanation of effects of assumptions on analytical approach good, minor mistakes made
VERY GOOD (90)	Reasonable number of assumptions and very good motivation Explanation of theory very good, no mistakes made Analytical results very good, no mistakes made Explanation of effects of assumptions on analytical approach very good, no mistakes made
EXCELLENT (100)	Reasonable number of assumptions and exemplary motivation Exemplary explanation of theory Excellent analytical results Exemplary explanation of effects of assumptions on analytical approach

NUM. MOD. (30%) 100 / 100

Numerical model

MISSING (0)	(Updated) flow chart for numerical approach missing and/or assumptions numerical approach missing and/or theory numerical approach and/or numerical results missing and/or effect of assumptions on results of numerical approach missing
INSUFFICIENT (40)	Flow chart of for numerical approach difficult to read and/or containing mistakes, revision of flowchart required Insufficient number of assumptions and/or too many assumptions and/or mistakes in the assumptions made Explanation of theory incomplete or with considerable mistakes Numerical results incomplete or with considerable mistakes Explanation of some of the effects of assumptions on numerical approach missing or with considerable mistakes
SUFFICIENT (60)	Flow chart of for numerical approach somewhat difficult to read/or containing quite some mistakes, minor revision of flowchart required Reasonable number of assumptions, but quite some mistakes in the assumptions made Explanation of theory sufficient, but with quite some mistakes Numerical results sufficient, but with quite some mistakes Explanation of effects of assumptions on numerical approach sufficient, but with quite some mistakes
MORE THAN SUFFICIENT (70)	Flow chart of for numerical approach easy to read but with some mistakes. Flow chart could be elaborated further Reasonable number of assumptions and more than sufficient motivation Explanation of theory more than sufficient, but with room for improvement, some mistakes made Numerical results more than sufficient, but with room for improvement, some mistakes made Explanation of effects of assumptions on numerical approach more than sufficient, but with room for improvement, some mistakes made
GOOD (80)	Flow chart of for numerical approach easy to read and with only minor mistakes. Reasonable number of assumptions and good motivation Explanation of theory good, only minor mistakes made Numerical results good, only minor mistakes made Explanation of effects of assumptions on numerical approach good, only minor mistakes made
VERY GOOD (90)	Very good flow chart of for numerical approach, only very minor comments can be made Reasonable number of assumptions and very good motivation Explanation of theory very good, no mistakes made Numerical results very good, no mistakes made Explanation of effects of assumptions on numerical approach very good, no mistakes made
EXCELLENT	Exemplary flow chart of for numerical approach Reasonable number of

(100)	assumptions and exemplary motivation Exemplary explanation of theory Excellent numerical results Exemplary explanation of effects of assumptions on numerical approach

VERIFICATION (15%) 90 / 100

MISSING (0)	Verification of individual code modules missing and/or verification of full model with a simplified model missing
INSUFFICIENT (40)	Verification of individual code modules insufficient and/or with considerable mistakes Verification of full model with a simplified model insufficient and/or with considerable mistakes
SUFFICIENT (60)	Verification of individual code modules sufficient, but with quite some mistakes Verification of full model with a simplified model sufficient, but with quite some mistakes
MORE THAN SUFFICIENT (70)	Verification of individual code modules more than sufficient, but with room for improvement, some mistakes made Verification of full model with a simplified model more than sufficient, but with room for improvement, some mistakes made
GOOD (80)	Verification of individual code modules good, minor mistakes made Verification of full model with a simplified model good, minor mistakes made
VERY GOOD (90)	Verification of individual code modules very good, no mistakes made Verification of full model with a simplified model very good, no mistakes made
EXCELLENT (100)	Exemplary verification of individual code modules Exemplary verification of full model with a simplified model

VALIDATION (20%) 100 / 100

MISSING (0)	Validation of full model with reference data missing and/or discussion/explanation of differences missing and/or recommendations for improvements of simulation missing and/or discussion whether right theory is chosen missing
INSUFFICIENT (40)	Validation of full model with reference data insufficient and/or with considerable mistakes Discussion/explanation of differences insufficient and/or with considerable mistakes Recommendations for improvements of simulation insufficient and/or with considerable mistakes Discussion whether right theory is chosen insufficient and/or with considerable mistakes
SUFFICIENT (60)	Validation of full model with reference data sufficient, but with quite some mistakes Discussion/explanation of differences sufficient, but with quite some mistakes Recommendations for improvements of simulation sufficient, but with quite some mistakes Discussion whether right theory is chosen sufficient, but with some mistakes
MORE THAN SUFFICIENT (70)	Validation of full model with reference data more than sufficient, but with room for improvement, some mistakes made Discussion/explanation of differences more than sufficient, but with room for improvement, some mistakes made Recommendations for improvements of simulation more than sufficient, but with room for improvement, some mistakes made Discussion whether right theory is chosen more than sufficient, but with room for improvement, some mistakes made

GOOD (80)	Validation of full model with reference data good, minor mistakes made Discussion/explanation of differences good, minor mistakes made Recommendations for improvements of simulation good, minor mistakes made Discussion whether right theory is chosen good, minor mistakes made
VERY GOOD (90)	Validation of full model with reference data very good, no mistakes made Discussion/explanation of differences very good, no mistakes made Recommendations for improvements of simulation very good, no mistakes made Discussion whether right theory is chosen very good, no mistakes made
EXCELLENT (100)	Exemplary validation of full model with reference data Exemplary discussion/explanation of differences Excellent recommendations for improvements of simulation Excellent discussion whether right theory is chosen

MISSING DATA (5%) 60 / 100

MISSING (0)	Discussion on how to deal with missing data not present in the report.
INSUFFICIENT (40)	Discussion on how to deal with missing data insufficient or erroneous.
SUFFICIENT (60)	Discussion on how to deal with missing data just sufficient or with quite some mistakes.
MORE THAN SUFFICIENT (70)	Discussion on how to deal with missing data more than sufficient or with some mistakes.
GOOD (80)	Discussion on how to deal with missing data good or with minor mistakes.
VERY GOOD (90)	Discussion on how to deal with missing data very good or without mistakes.
EXCELLENT (100)	Exemplary discussion on how to deal with missing data.