Faculty of Aerospace Engineering

Thesis Report

Finding the trim points of the ICE model using interval analysis

S.C. Hungs

January 20, 2018

TU Delft Delft University of Technology

Challenge the future

Thesis Report

Finding the trim points of the ICE model using interval analysis

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

S.C. Hungs

January 20, 2018

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Copyright © S.C. Hungs
All rights reserved.

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"Thesis Report"** by **S.C. Hungs** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: January 20, 2018

Readers:

dr.ir. E. van Kampen

dr.ir. Q. P. Chu

dr.ir. C. C. de Visser

dr.ir. W. van der Wal

Acronyms

AMT	all-moving wing tip
DLEF	differential leading edge flap
EoM	equations of motion
ICE	Inovative Control Effectors
LOC-I	loss of control in-flight
PF	pitch flap
PTV	pitch thrust vectoring
SQP	sequential quadratic programming
SSD	spoiler-slot-deflector
YTV	yaw thrust vectoring

List of Symbols

Greek Symbols

α	Angle of attack
β	Angle of sideslip
δ_{lamt}	Left all-moving wing tip deflection
δ_{lele}	Left elevon deflection
δ_{llefi}	Left inboard leading edge flap deflection
δ_{llefo}	Left outboard leading edge flap deflection
δ_{lssd}	Left spoiler-slot-deflector deflection
δ_{pf}	Pitch flap deflection
δ_{PTV}	Pitch thrust vectoring deflection
δ_{ramt}	Right all-moving wing tip deflection
δ_{rele}	Right elevon deflection
δ_{rlefi}	Right inboard leading edge flap deflection
δ_{rlefo}	Right outboard leading edge flap deflection
δ_{rssd}	Right spoiler-slot-deflector deflection
δ_{YTV}	Yaw thrust vectoring deflection
ϕ	Body roll angle
ψ	Body yaw angle
ρ	Air density
θ	Body pitch angle

Roman Symbols

a	Speed of sound
b	Wingspan
$\bar{c}$	Mean chord
C_L	Moment coefficient about the X-body axis
C_M	Moment coefficient about the Y-body axis
C_N	Moment coefficient about the Z-body axis
C_X	Force coefficient along the X-body axis
C_Y	Force coefficient along the Y-body axis
C_Z	Force coefficient along the Z-body axis
D	Down component of NED tangent plane
E	East component of NED tangent plane
g	Gravity acceleration
M	Mach number
N	North component of NED tangent plane
p	Body rotational velocity about x-axis
q	Body rotational velocity about y-axis
$\bar{q}$	Dynamic pressure
r	Body rotational velocity about z-axis
T	Thrust
u	Body velocity in x-axis
V	Airspeed
v	Body velocity in y-axis
w	Body velocity in z-axis
x	Crisp number named 'x'
$\mathbf{x}$	Crisp vector named 'x'
$\mathbf{X}$	Crisp matrix named 'x'
$[x]$	Interval number named 'x'
$[\mathbf{x}]$	Interval vector named 'x'
$[\mathbf{X}]$	Interval matrix named 'x'

Contents

Chapter 1

Introduction

Aviation safety is the industry's biggest interest. Any research aiming to improve safety can be considered to be a good investment. What that research should focus on is not always obvious, but statistics can provide good guidance in the matter. The annual safety report from the ICAO shows that in 2015 10% of all fatalities was due to high-risk accident occurrence categories: runway safety related events, loss of control in-flight (LOC-I), and controlled flight into terrain (ICAO, 2016). Although 10% does not seem like a lot, please remember that 2015 saw two mayor accidents that fall out of these categories: Germanwings Flight 9525 which was a deliberate crash from a suicidal copilot and Metrojet Flight 9268 for which the research is still ongoing. It is however suspected that it was caused by a bombing (Georgy & Evans, 2015). With 150 and 224 fatalities respectively, these two accidents make up 79% of all fatalities in 2015. When not taking these two accidents into account the high-risk categories make up 47% of the fatalities.

When looking at the number of fatal accidents, so not the number of fatalities, it can be seen that the high risk categories make up half of all accidents. More specifically, when comparing the high risk categories it turns out that LOC-I is the main contributor with 33% of all fatal accidents. This shows that it is worthwhile to invest in research trying to prevent LOC-I. A study has been performed to do nonlinear analysis on LOC-I (Kwatny et al., 2013). This study shows that loss of control "*is generally associated with flight outside of the normal flight envelope, with nonlinear influences, and with a significantly diminished capability of the pilot to control the aircraft*" (Kwatny et al., 2013). During the quantification of LOC-I five two-dimensional envelopes are used to find the safe set of states and control inputs, where the safe set is defined as a positively invariant subset of those envelopes. When finding the safe set a bifurcation method is applied on a set of trim points. Because trim is defined as steady motion with control inputs that do not exceed the control limits the trim set will always be part of the safe set.

A different effort to prevent loss of control accidents is the determination of the safe maneuvering envelope (Van Oort, Chu, & Mulder, 2011). In this paper a reachability analysis has been applied on a nonlinear high-fidelity F-16 aircraft model. By starting from an a-priori known safe set the backwards and forwards reachable sets are determined. The backwards

reachable set is defined as all the states for which a control strategy exists that returns the state back into the safe set, while the forward reachable set is defined as all the states for which a control strategy exists that brings the state from the safe set into that state. The intersection of these two sets is the safe operating set. Again, by definition the trim set can be used as the a-priori safe set.

Both of these cases show that it is important to know the trim set of an aircraft while determining its safe operating envelope. Trim points can also be used for different purposes. First of all flight control systems use trim points as initial conditions from which the internal model of the aircraft is linearized. Because an aircraft is flying in different configuration depending on the phase of flight, different trim points must be known to develop a proper controller for every part of flight. Knowing the trim points of an impaired aircraft can also help to come up with fault tolerant design. Secondly trim points can be used for flight simulators. Depending on the training scenario the initial conditions are usually trim points of the aircraft model (De Marco, Duke, & Berndt, 2007).

This research will find the trim set of the Inovative Control Effectors (ICE) model developed by Lockheed Martin. This model has thirteen control effectors and a lot of coupled dynamics, making it particularly challenging to find the full trim set. To do so interval analysis will be used. Interval analysis is a branch and bound method which ensures that all minimums will be found when solving minimization problems. Branch and bound methods are generally computationally heavy, so there is a big challenge on hand to use available computing power effectively. The second mayor challenge comes from the ICE model itself. There is redundancy in the control effectors, which can result in trim points running into one another creating trim lines, or even trim surfaces or (hyper)volumes. To find them is not so much of a problem when using interval analysis, but to then present them in a meaningful and understandable way is a different story.

As mentioned, the aim of this research is to find the trim set of the ICE model. This can be converted into the main research question: how can the trim points of the ICE model be found using interval analysis? This question contains three main parts that can lead to subquestions. These are listed below.

1. What are trim points?
2. What is the ICE model?
3. Why is the ICE model a suitable subject for this research?
4. Why is interval analysis a suitable method for finding aircraft trim points?
5. What makes interval analysis especially useful as a method for finding the trim points of the ICE model?
6. How should an interval analysis routine be implemented to be able to find trim points?
7. What is the trim set of the ICE model?
8. How can the trim set be presented in a useful way?

After this introduction the most important results will be presented in a AIAA paper format. Chapter 2 will continue with an extended description of the ICE model. Next is Chapter 3, which will give an overview of the trim problem and what methods have been used to solve it. An introduction to the basic principles of interval analysis, along with an outline of how interval optimization works is given in Chapter 4. Chapter 5 presents results of some interval analysis optimizations, showing that it is indeed a good solver. Some different implementations of parts of the interval routine are compared as well. After that Chapter 7 will give additional results that have not been discussed in the paper. Conclusions are given in Chapter 8. That chapter will also contain recommendations on how to follow up on this research.

Finding the trim set of the ICE model using interval analysis

Stephen C. Hungs
Delft University of Technology, Delft, Zuid Holland, 2629 HS, The Netherlands

In this work interval analysis is applied to the thirteen control effector Innovative Control Effectors model to find its trim set. The method to find trim states is based on interval box consistency. At low speed the method is capable of finding interval enclosures of single trim points with a high accuracy if the minimum number of required control effectors is used. At higher speeds the found accelerations are larger. When looking for a full trim set the method finds continuous bounds on the control effectors for the entire input range in one run. This is a good demonstration of the advantages that interval analysis has over conventional methods that generally can only find one trim point at a time. The found bounds are a maximum of 1° wide for each control effector, but despite this the remaining accelerations can be up to $0.5m/s^2$ for linear accelerations and up to $10^\circ/s^2$ for rotational accelerations. Because of these large accelerations the found solutions are not acceptable as trim conditions. On the other hand the potential that interval analysis has as a trimming method is demonstrated, since continuous bounds on trim sets have been found in a single run. This is a feat that no other trimming method has yet accomplished. Further research is needed to exploit the full potential of interval trim methods so that the results can be used for other purposes such as flight envelope prediction.

I. Introduction

The most frequent cause of fatalities in aviation is loss of control in-flight (LOC-I) [1]. It has been shown that LOC-I "*is generally associated with flight outside of the normal flight envelope, with nonlinear influences, and with a significantly diminished capability of the pilot to control the aircraft*" [2]. In that research a bifurcation method is applied to a set of trim points to identify the normal flight envelope. A different approach to finding the safe flight envelope relies on the intersection of the forwards and backwards reachable sets, starting from a safe set [3]. Again a trim set is used as an initial safe set. This shows the importance of knowing the trim set of an aircraft to determine safe operating limits. The reason that trim sets can be used as a safe set is that a trim point is a dynamic equilibrium state of the aircraft. This means that the aircraft will maintain that state indefinitely, which can be considered a safe situation.

This research focuses on the innovative control effectors (ICE) model developed by Lockheed Martin. The main purpose of the ICE model was "*to investigate the potential for new and innovative methods for stabilization and control for high performance, low all-aspect signature fighters*" [4]. The ICE model is an interesting model to try trim methods on because it has a total of thirteen control effectors. A consequence of this is that there is redundancy in control power, potentially leading to trim curves or even surfaces instead of points for given flight conditions.

Aircraft trim can have various meanings depending on the context. In this work trim is considered to be all the states of the aircraft model where the accelerations are equal to zero. The trim problem is the task of finding those states, and is generally solved with optimization methods. Interval analysis is a branch and bound optimization method that is capable of computing bounds on all possible solutions in one single run [5], [6]. Although it is a computationally heavy method it has been shown to have potential as an aircraft trim method [7], as it found both the normal side and the back side of the power curve in one run. This makes it a very strong method compared to other methods that have been tried in the past, such as gradient methods [8], sequential quadratic programming [9], bifurcation and continuity analysis [10] and [11] and evolutionary algorithms [12]. All these methods find only one of the possible solutions, based on what the initial conditions were. An analytical solution to the trim problem was proposed in [13], but in order for this to work the model had to be linear in the control inputs. This limits its usability. Another strong feature of interval analysis is that it is very easy to enforce constraints on a problem [5], [6]. Applying constraints is almost unavoidable when dealing with an aircraft model that has thirteen control effectors.

First a short introduction to interval analysis will be given in Section II. The ICE model is discussed in Section III. In Section IV the approach to solving the trim problem will be explained. This is followed by the results in Section V. This paper is ended by conclusions and recommendations in Section VI.

II. Interval Analysis

Interval analysis is a field of mathematics that deals with intervals instead of crisp numbers. Each number is represented by a lower bound a and an upper bound b to make the interval $[x] = [a, b]$. Initially it was developed to put bounds on rounding errors that originate from the limited precision of numbers on digital computers [14], but it was soon realized that interval analysis could be used as an optimization tool as well [15]. If for example 64-bit floating point arithmetic is considered in accordance with IEEE Standard 754 [16], the smallest number greater than 1 is $1 + z$ where $z = 2.220446049250313 \cdot 10^{-16}$. With normal computation the result of $1 + \frac{z}{3}$ would be simply 1, but with interval arithmetic it would be $[1, 1 + z]$. Neither the lower or the upper bound are the exact solution, but the exact solution is certain to be contained by the interval solution. Normal math operations can be done on intervals, but this requires that the rules are extended. If any operator is expressed as $\bullet$ then [5]:

$$[x] \bullet [y] = \{x \bullet y | x \in [x], y \in [y]\} \tag{1}$$

For example, refer to Equations 2 and 3 to see what this means for addition and multiplication.

$$[a, b] + [c, d] = [a + c, b + d] \tag{2}$$

$$[a, b] \cdot [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)] \tag{3}$$

When intervals are used as a function argument the function value will also be an interval. Care must be taken that the function has as few occurrences as the interval variable as possible. As an example take Equations 4 to 6. In crisp arithmetic these expressions would be equivalent. If now the interval $x = [-1, 3]$ is used instead they evaluate to 3 different intervals. This problem is known as interval dependence. It is certain that all possible values that the function assumes on the argument interval are contained by the resulting interval. If the function is expressed without taking dependence into account there will be overestimation in the function evaluation. When every variable is in the function expression only once the interval will be tight, meaning that the lower bound equals the minimum value of the function on the domain and the upper bound the maximum value. This is the case in Equation 6.

$$[x]^2 - [x] = [-3, 10] \tag{4}$$

$$[x]([x] - 1) = [-6, 6] \tag{5}$$

$$\left([x] - \frac{1}{2}\right)^2 - \frac{1}{4} = [-0.25, 6] \tag{6}$$

In [5] approaches to optimization algorithms based on interval analysis are explained. Used in such a way interval analysis is a very powerful tool, because it finds all possible solutions in one single run. This feature is shown to be extremely useful in for example parameter estimation, robust control and robotics [6] or for nonlinear aircraft trim, optimization of human perception modeling and spacecraft rendezvouz and docking [17]. It has also been used to evaluate the effect of uncertainties for re-entry flight clearance [18]. Interval analysis does have downsides, such as the computational load associated with it. By definition it will not find the exact solutions, but only bounds on the solutions. However, for practical purposes the mid point of the interval can be considered a crisp solution if the bounds are tight enough. When considering intervals as inputs to an optimization problem they are generally called boxes. In multidimensional problems a box consists of a vector of intervals. All calculations in this work are done with Matlab in combination with the Intlab toolbox [19]. The implementation of the optimization algorithm will be discussed later in Section IV.

III. Model

The ICE model was developed by Lockheed Martin as an effort to reduce the signature of fighter aircraft, while still being highly agile. To this extend the vertical tail has been removed and instead three promising alternative concepts for yaw control have been applied to the model [4], [20]. The three concepts are all moving wing tips (AMT), spoiler-slot deflectors (SSD), and differential leading edge flaps (DLEF). Combined with elevons, a pitch flap (PF), pitch thrust vectoring (PTV) and yaw thrust vectoring (YTV) this aircraft has a total of thirteen control effectors. There are two

versions of it, a delta wing shaped model for the air force (configuration 101) and a diamond wing with canard layout for carrier based use for the navy (configuration 201). An impression of the 101 model is given in Figure 1. Wind tunnel tests on the 101 model have shown nonlinear characteristics of the aerodynamics and in some cases even discontinuities [21].

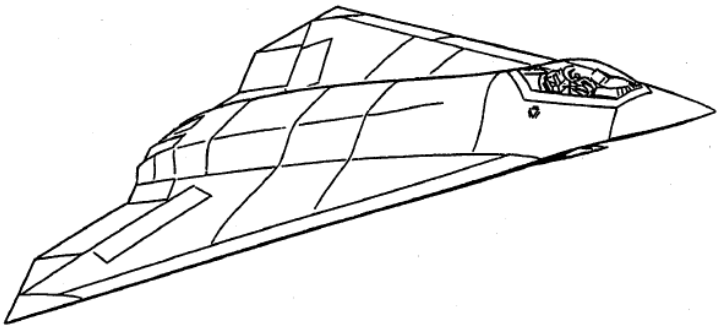

Fig. 1 Impression of what the 101-series ICE model looks like [4]

The model of the 101 configuration of ICE is written as a Simulink model. Unfortunately the interval toolbox Intlab is not compatible with Simulink, and because of that the model must be converted to a Matlab code equivalent. This has been taken as an opportunity to write the model specifically with trim in mind. All states in the original model that are used as feedback states are now linked directly to the input. Based on that expanded input set the aerodynamic coefficients for forces and moments can be calculated. These are then used to determine the aerodynamic body forces and moments. Because the converted model only will be used for trim the thrust setting can be coupled to the body forces. It simply needs to be equal to the body force in x-direction. The aerodynamic model is not symmetrical. This causes problems if straight and level flight conditions are assumed for trim. In those cases the pitch angle equals the angle of attack and the roll angle should be zero degrees. When zero degree roll angles are used the asymmetry will result in acceleration intervals of the model not containing zeros. As a consequence of that no trim solutions will be found. To counter this the aerodynamic side force can be used as a measure for how large a roll induced gravitational side force must be. In this way roll angles can be found that do result in a trimmed state. Because the mass and the inertia of the model are constant the accelerations can easily be calculated from the forces and moments. All flight states required to do the calculations are used as an input or depend on the inputs directly. This removes the need to do integration in the model. This is a good thing because integration generally produces very wide intervals [22], [23].

The conversion has changed the list of inputs to the model. In the original model only the control deflections and the thrust setting could be controlled directly, while some flight parameters such as initial altitude and velocity could be set before a simulation. The states of the model were all calculated internally, based on integration from the initial values. The outputs were the states, with the speeds expressed in earth velocities, body velocities, and α, β, Mach representation. Instead of Mach number the airspeed can be used, as that is calculated too. Finally the dynamic pressure is given as an output as well.

In the converted model the thrust setting is internalized and can thus be removed from the list of inputs. On the other hand a lot more control on the flight state is desired, so that trim for specific scenarios can be found. Because of that the angle of attack, angle of sideslip and the Mach number are usable as inputs now. Table 1 gives an overview of the inputs and their allowable ranges, as well as the sign conventions used in this work. The choice of flight condition determines how the Euler angles need to be linked to the inputs. This means they can be internalized as well. The flight condition also dictates the values for the body rates. Since trim is the goal, it is known that the accelerations must be zero, so constant values can be used for the rates. The same holds for the altitude. The position on earth is arbitrary in this model. This means that the only the accelerations are missing. These are the outputs. In order to reconstruct the power curve thrust is required as an output as well.

The aerodynamic model of ICE is stored in 108 data tables. Of those 108 there are 96 tables that use linear interpolation and twelve that use cubic interpolation. Linear interpolation with intervals can be done by evaluating a finite number of points and taking the maximum and minimum value at those points as upper and lower bounds [7]. These points are every data point contained by the box, all points on the boundary of the box intersecting a line between data points and all corners of the box. Figure 2 shows this.

Table 1 Allowable ranges and sign conventions for the inputs to the converted model

Input	Lower limit	Upper limit	Sign convention positive
α	$-2.5°$	$42.5°$	Nose up
β	$-30°$	$30°$	Nose right
Mach number	0.3	2.16	Positive body x-axis
AMT	$0°$	$60°$	Trailing edge down
DLEF inboard	$0°$	$40°$	Leading edge down
DLEF outboard	$-40°$	$40°$	Leading edge down
Elevons	$-30°$	$30°$	Trailing edge down
PF	$-30°$	$30°$	Trailing edge down
SSD	$0°$	$60°$	Trailing edge up
PTV	$-15°$	$15°$	Nozzle down
YTV	$-15°$	$15°$	Nozzle left

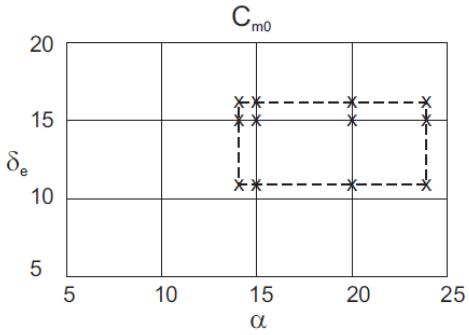

Fig. 2 All points that need to be considered during linear interpolation in an interval box [7]

The process for cubic interpolation is a bit more elaborate. The explanation of how and why it works is as follows. The second derivative of a cubic spline between two data point is at most a linear function. Therefore there can be at most one zero crossing for the second derivative, hence there can be at most one inflection point between two data points. If the tangent is taken at the data points and extended to the other side of the section an inclusion interval can be determined. When there are no crossings of the tangent and the curve the endpoints of the tangent are measures for the inclusion interval of the curve. If the curve intersects the tangent line before it reaches the other side of the section this means that there has to be an inflection point between the data point at which the tangent was taken and the intersection of the tangent and the curve. This means that the curve can only continue in the direction that it has crossed the tangent. If the tangent is taken from the left data point it runs from point the data point A to the mapped point B. The tangent from the right data point C runs to the mapped point D. It is certain that the curve is included in the interval $[min(A, B, C, D), max(A, B, C, D)]$. The left plot in Figure 3 shows this procedure. When this is done over all the sections defined by data points and domain boundaries an inclusion interval for the entire domain can be found by taking the maximum and minimum found values, as is illustrated in the right plot of Figure 3. The curves used to make the figures are based on a cubic spline created on a set of ten random numbers in the range $[0, 1]$ randomly distributed on the domain $[0, 5]$. Because it uses the gradient to find the inclusion interval this interpolation method is called the cubic gradient inclusion interval. The current implementation used in the model determines the gradient numerically by taking the symmetric derivative around the data points.

After the conversion the two versions of the model are compared. This is to validate the new model. To do this a set of 10,000 randomly generated input combinations was used. All inputs were within the ranges indicated in Table 1. Based on each combination of inputs both models are run in a block by block fashion, to identify mismatches. During this process the largest differences were six orders of magnitude smaller than the actual values, while most of them were

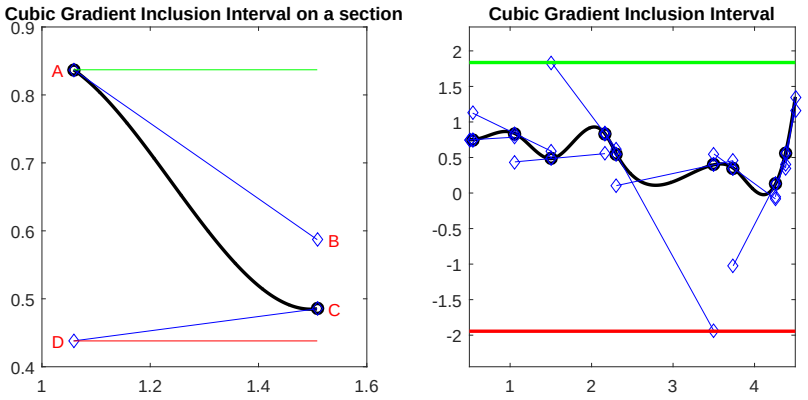

Fig. 3 Illustration of the workings of the cubic gradient inclusion interval

in the order of 64-bit double machine precision. The larger differences generally occurred at large aerodynamic angles, when the control surface deflections are large and disordered.

IV. Methodology

The process of finding a trim set starts with defining the flight condition and what control effectors can be used to steer the aircraft. Control effectors that are not allowed to move get assigned a fixed value, as do the flight states that do not change. The rest of the variables are independent variables. These get assigned an interval in which they are allowed to change. The altitude and the rates are in the current version not available to use as independent variables. This leaves α, β, and M as potential independent variables. Every control effector can be used as an independent variable as well. The sixteen element vector consisting of α, β, M and the control effector deflections is defined as a box. The input values set the search domain and make up the first box. The trim routine itself is basically a zero finding algorithm that is applied to the ICE model. An interval zero finding algorithm works as follows. To start the first box is evaluated. If there are zeros in this box there could be actual zeros in the function that is being optimized. If this is the case the initial box is split into subboxes. The function is evaluated again over these subboxes. If there are no zeros in a subbox it is certain that there are no zeros in the function for that part of the domain so the code can discard that box. The boxes that do have zeros can be split again, and the above process is repeated. This process is based on box consistency as explained in [5] and [24]. In box consistency the domain is split up in smaller pieces, for which it is ensured that the function values are consistent with the solution of the optimization problem. To visualize this process refer to Figure 4. Here a one dimensional function is evaluated over 20 subboxes. Of those boxes there are only six that might contain zeros. For the other fourteen it is certain that the do not contain zeros, and are therefore removed for further evaluation. The six remaining boxes are each split into five smaller boxes and after evaluation eight boxes remain. If this process is repeated several times the bounds on the location of the zeros can become very tight. For the trim algorithm each split will happen in a different box direction, until all box directions have been split once. Then the code splits the remaining boxes again in the first direction and loops through all directions until all are split once more. This goes on until the stopping criteria are met or until there are no boxes remaining. The stopping criteria are based on practical limitations and limitations of the human vestibular system. The practical limitations come from state measuring accuracy and control deflection precision. There is no point in knowing how much the accelerations vary over a box that spans 0.00001° in angle of attack, because the angle of attack sensor at the actual aircraft is not able to measure that accurate. If boxes are split so often that they become smaller than 0.01° in angle of attack or angle of sideslip and smaller than 0.005 in Mach number the splitting is terminated in that direction. As will be explained below, it is possible to do trim runs focused on finding point enclosures and runs aiming to find the entire trim set on a domain. It can be expected that point enclosures do not require as much boxes as trim set enclosures. Therefore runs aimed at trim points get a sharper control deflector box diameter stopping criterion of 0.01°. Box splitting in trim set runs terminate if the control deflector box diameter is smaller than 0.1°. The limiting factor from the human vestibular system is the limit to what it can actually detect. From [25] it is found that the smallest accelerations that can be detected are $0.02 m/s^2$ for linear accelerations and $0.05^\circ/s^2$ for rotational accelerations. These values indicate when the human sensory system can no

longer detect accelerations in a controlled simulator environment. If either of the stopping criteria are met the code terminates. If the box precision is reached the found results should first be evaluated before going further, due to the computationally demanding nature of interval analysis. If the number of boxes starts to get excessive there it can become a very slow process that hardly gives any improvements. If on the other hand the sensory limits are met before the box precision is achieved this means that the entire box can be used as trim input. No further refinement is necessary in that case. An overview of the process is given in Figure 5.

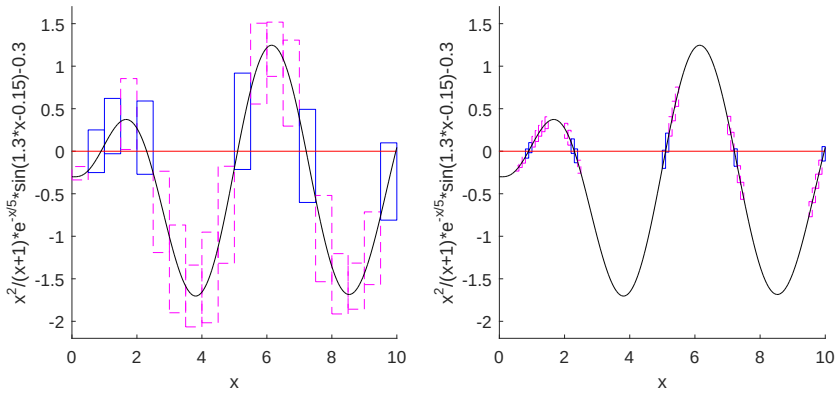

Fig. 4 Visualization of an interval analysis zero finding algorithm on an example function

With thirteen control effectors the dimensionality of the ICE model is much to large to solve in a single run with the current state of technology. The flight condition and limiting the number of active control effectors can help in the matter. Some of the more useful flight conditions are explained in [8]. From this it is known that choosing for steady straight and horizontal flight pins down quite some variables. The altitude is simply a constraint, while the rates are fixed at zero. It is also known that the pitch angle θ must be equal to the angle of attack α. As explained earlier there is an asymmetry in the ICE model that requires a nonzero roll angle ϕ in order to achieve a trimmed condition. Because of this it is not sufficient to simply use the elevons for pitch control and trim the aircraft like that. From [4] and [20] it is known that the AMTs perform reasonably constant up until an angle of attack of 40°. At even higher angles the control power reduces, but it is still sufficient to keep full body control. AMTs can provide yaw control as well as roll control. Yaw control is linear with control deflection for a large range of angles of attack. Roll control is not linear and its effect can even change from adverse at low angles of attack to favorable at high angles of attack. If this is combined with elevons the model has full roll, pitch, and yaw control. To keep the trim problem small a simple test is performed to see if trim can be achieved by using only one AMT. It turns out that only using the left AMT is sufficient. The initial interval for each control effector will be the full allowable range, to ensure that all possible trim points will be found. The above has explained the choices for determining what control effectors to use. Furthermore the asymmetry in the model is resolved by just using roll. With steady straight and horizontal flight the sideslip angle can be set at zero. If both Mach number and angle of attack are used as free variables every combination of Mach number and α that gives a trimmed state will be found. This approach will also produce bounds on the free control effectors. Before attempting such a run it must be ensured that the algorithm is working properly. To do this Mach number is used as a constraint. By checking the solutions at multiple Mach numbers trends can be determined. For example, the shape of the power curve should comply with the shapes of the power curves of other aircraft. This approach also enables comparison between the interval trim method and the conventional trim method that was provided with the ICE model. That method uses the Matlab function `fmincon` with the active-set algorithm. Comparing the results found with both methods gives an indication if the interval method converges to the right solutions. After confirming that the interval methods find the right solutions a run that finds every trim combination of Mach number and angle of attack can be executed. For both variables their full allowable range is used as the input to the solver.

V. Results

As explained above, it is important to ensure that the code actually finds the right solutions. To do that the first set of results is aimed at point solutions. Fifteen sample speeds are used to find these points. These are taken with

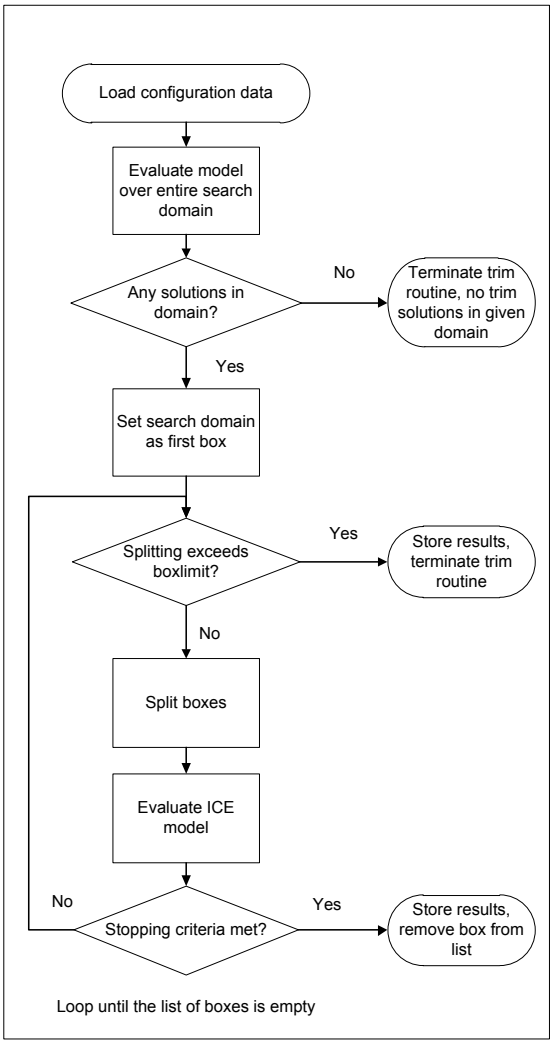

Fig. 5 Flowchart showing the process used for trimming the ICE model

equal spacing from the allowable Mach range, spanning the entire range from 0.3 to 2.16. Two different altitudes are used as well, one set at 10,000 ft and one at 20,000 ft. In total that means 30 runs are required to get this initial set of results. The code produces bounds on the true solutions, so the outcome is the list of boxes of input combinations that would result in interval accelerations that do contain zeros. To keep the plots of these results clear the mean value of the midpoints of the list of remaining boxes is taken. The hull of these lists is at most 0.02° wide in angle of attack, 0.12° wide in the control deflections and a maximum of 120 lbf wide in terms of thrust setting. Relative to the absolute values the hulls are very small, so the difference between a crisp point or the hull of the boxes would not be visible. The plots can be found in Figure 6. Individual plots are made for all the free variables of the zero finding problem, as well as the associated thrusts.

The plot of the thrust against the Mach number is widely known as the power curve. This can be used as a indication if the code converges to the right solutions, since every aircraft has one and the trends in power curves are generally the same. For each aircraft there is a velocity that requires a minimum amount of thrust. This is because the combination of induced and parasitic drag is the lowest at that point. Fly faster and the parasitic drag will increase, requiring more thrust. Flying slower than that point increases the induced drag, again increasing the required thrust. This region where more thrust is needed to fly slower is known as the backside of the power curve. The trends between the points in the first plot in Figure 6 show this behavior as well. The impact of different altitudes is clear as well. Because the atmosphere at lower altitude is denser the power curve is steeper. The point of lowest thrust also occurs at a lower speed.

This matches the trends seen in the plot. What also can be seen in the plot is that there are two points missing in the curve for 10,000 ft. A hint of what happened there can be seen in the plot for the angle of attack. The solution at Mach 1.3628 appears to be out of line with the rest of the solutions in that it finds a lower angle of attack compared to the ones at the higher Mach numbers. When allowing the code to exceed the allowable range in angle of attack as indicated in Table 1 to highly negative angles it does find a solutions. These solution require angles of attack of approximately $-15°$. When using the trim procedure that came with the model similar results are found. This indicates that there is nothing wrong with the method, but rather that there is some odd behavior in the model in that particular flight regime. The trend for the angle of attack curve at 20,000 ft shows the expected behavior of high angles at low airspeed with the curve flattening off to an almost constant angle at higher speeds.

When looking the required deflections at the trimmed points there are three parts to each curve: the part on the backside of the power curve, the part on the normal side of the power cure up to Mach 1.5, and the part that covers the higher sonic regime. In the first domain it is very notable that the required LAMT deflection is larger than it needs to be for the other parts of the speed domains. The elevon deflections at 20,000 ft show the expected reversal of command effect at the slowest speeds, which cannot be seen at 10,000 ft. The section that covers the part up to Mach 1.5 has only small LAMT deflections, with maximum deflections of $1.5°$, while the higher sonic part of the domain sees even smaller LAMT deflections up to $0.5°$. This indicates that the faster the aircraft flies, the more the LAMT acts as a trim tab. This is as expected, because in this scenario the LAMT is used just to balance the asymmetry of the model. It is also observed that the difference between the left and right elevon is smaller if the LAMT deflection is smaller. It can be assumed that if the model was symmetric, trim would be possible with just the elevons. Both elevons could use the same value in that case, leading to a very small optimization problem with only angle of attack and elevon deflection as free variables.

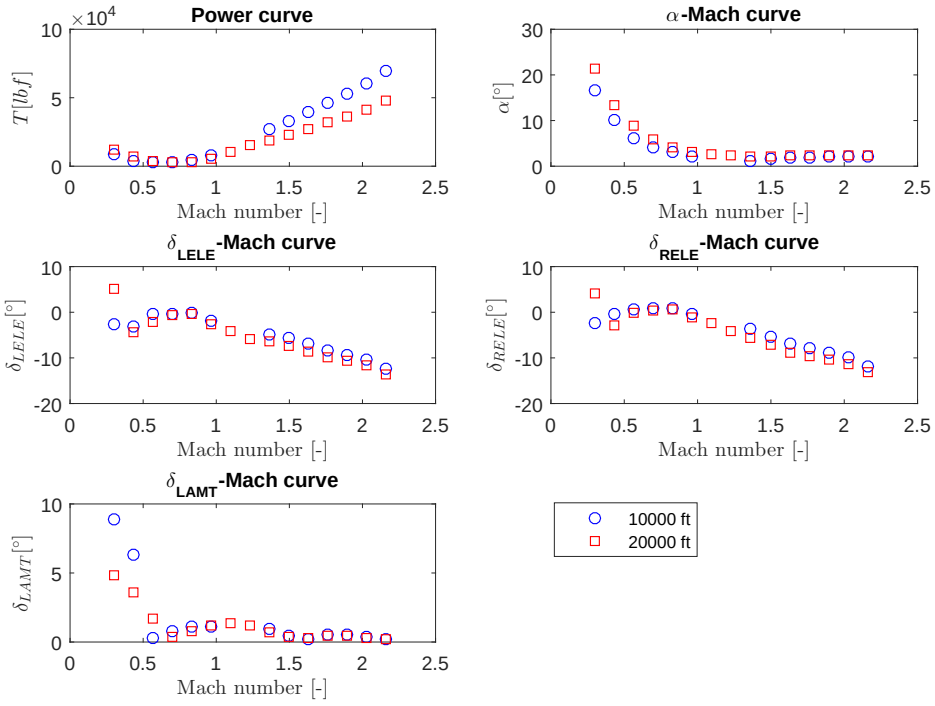

Fig. 6 Overview of the found trim points when using fixed Mach numbers

To compare the interval method with the more conventional trim method that came with the ICE model a similar set of runs is done with the standard trimming routine. At two different altitudes fifteen different velocities have been used to find a trim point. The found points are shown in Figure 7. For reference the points found with the interval method are given as well. As mentioned earlier the `fmincon` routine also finds odd results for Mach numbers 1.0971 and 1.2300 at 10,000 ft. Again, these points are not plotted, since this issue is rooted in the model itself. When comparing the two

sets of solutions one must keep in mind that the interval method only works on the converted model, which can only use intervals. This in turn means that the converted model cannot be used with the conventional trim method. A pure comparison between the methods is therefore not possible. But as mentioned in Section III the differences between the two versions of the model have been shown to be very small. This is notable at subsonic speeds, where the two methods find almost identical solutions. Going to higher speeds there are only minor differences in terms of thrust at 20,000 ft in the area where 10,000 ft runs do not find any solutions at all and at the highest Mach number that has been evaluated. In angle of attack there are no significant differences. In the control deflections there are bigger differences that can be as large as 3°. The only logical explanation is that these differences are caused by the combination of each model with each respective model. The differences between the models have been shown to be very small, so only part of the mismatch is caused by the cumulative mismatches of the individual parts of the models. The rest should therefore originate from the different features of each method. The interval method simply tries to bound zeros in the model accelerations, irrespective of the control surface deflections, while the `fmincon` code minimizes the sum of the squares of all control surface deflections. That the accelerations should be zero is only used as a constraint. This makes in logical that the absolute deflections with the `fmincon` method are smaller than the ones found with interval analysis. It must be noted that the `fmincon` code was not able to find a solution for Mach 1.4957 at 10,000 ft and for Mach 1.6286 at both altitudes with the used starting values of all control deflections equal to zero with an angle of attack of 5°. However, by changing both initial elevon deflections to −20° a solution is found for the 20,000 ft case. This point is indicated with a different marker in Figure 7. That the initial values can cause the method to not find a solution is a clear downside of the method. Interval analysis does not have this issue, as the only required input is the search domain.

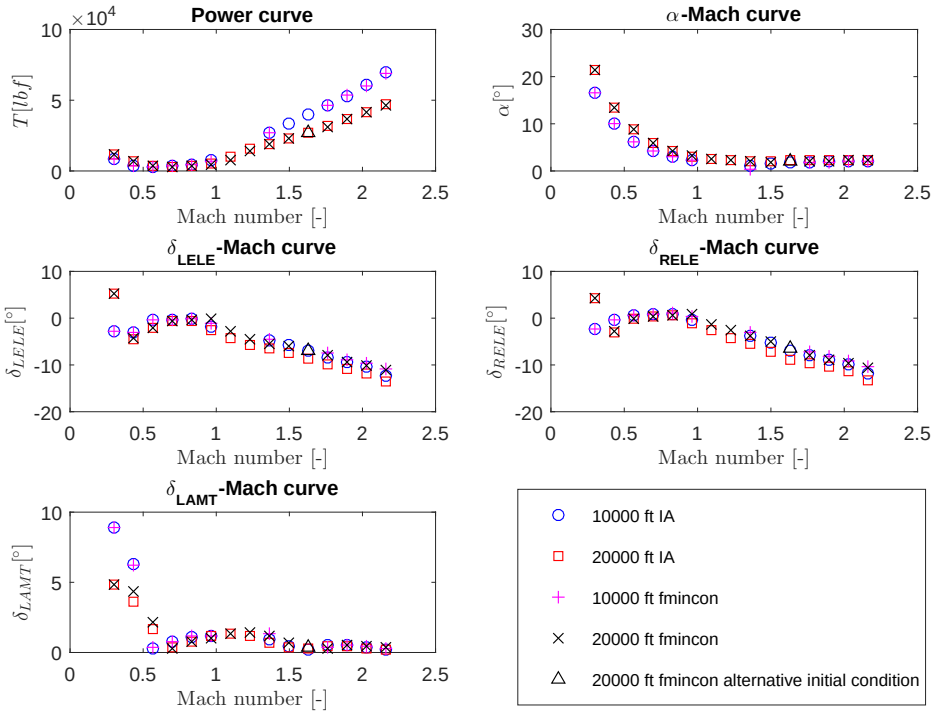

Fig. 7 Differences in the found results when using interval analysis and fmincon

The point solutions in Figure 6 have indicated that the trim routine is working as intended. It can now be expanded to also use Mach number as an independent variable. Two runs are done, one to cover each altitude that was used to get the previous results. The outcome of such a run is a set of continuous bounds on the free variables. These are shown in Figure 8. In terms of thrust the bounds are less than 500 lbf apart for the most of the domain, only at high Mach numbers where high thrust settings are needed the difference goes up to a maximum of approximately 900 lbf. The upper and

lower bounds on angle of attack are rarely more than 0.2° apart, while the control deflectors bounds are on average 0.37° wide, exceeding 1.0° only once. The points found previously are all contained by the bounds found in these runs. As with the point solutions, there is a domain where the model shows some odd behavior and requires highly negative angles of attack for trim. This can best be seen in the α-Mach curve, where the bounds on the angle of attack suddenly drop to the lower limit of the validated range, only to gradually recover at higher velocities. In these sections the bounds on thrust and angle of attack are further apart than the limits stated before. In the plots there are several kinks, such as the one at Mach 1.2 in the thrust plot and the ones at Mach 0.9 and Mach 0.95 in the elevon plots. These kinks happen at data points in the tables. Because most data tables interpolate linearly there are discontinuities in the derivatives of the data values. This carries through to the found solutions. Except for the gap in the 10,000 ft run the power curves look as they are supposed to look. The effect of altitude can again clearly be seen with the curve less steep and moved to the right. The higher density of the atmosphere at the lower altitude means that less angle of attack is needed to generate enough lift, which can also be clearly seen. The continuous bounds on the control effectors show that there is a lot more variation going on in the required deflections on the backside of the power curve than the point solutions initially suggested. The two slowest solutions on the LAMT plot in Figure 6 might have suggested that more LAMT deflection is needed at lower altitudes, but as it turns out the required deflections have a similar profile, with the one for greater altitude shifted to the right the same amount as the power curve is shifted. The fact that the point solutions of the 10,000 ft run were close to local maximums and the points for the 20,000 ft run close to local minimums gave an unclear picture of the true deflection profiles. Both elevon profiles show similar right shifting with increasing altitude. As with the point solutions, the difference between the left and right elevon deflection is smaller when the LAMT deflection is smaller. This enforces the hypothesis that the model could be trimmed with only elevons if it was symmetric.

In addition to the five plots that are also there in the point solution figure, there are two plots showing information about the branch and bound algorithm as it progressed to the solution. The one on the left gives the number of boxes that could potentially contain trim points after each step. The computer that was used during these calculations could evaluate an average of 160 boxes per minute. Because the splitting algorithm divides the remaining boxes into three subboxes the cumulative number of boxes for all iterations but the last must be multiplied by three to get the number of boxes that have been evaluated. What this translates to is that the 10,000 ft run took about 6 days and 10 hour to converge. The run at 20,000 ft used more boxes, which lead to a total calculation time of approximately 7 days and 8 hours. The runs enclosing trim points required significantly fewer boxes and because of that a lot less time too. For all 15 runs at 10,000 ft only 3 hours and 22 minutes were needed. In this case the 20,000 ft runs used more boxes too, leading to a calculation time of 3 hours and 37 minutes. Because the high number of boxes and the associated computation time was anticipated the control deflection box diameter stopping criterion was stretched from 0.01° to 0.1° for the runs finding continuous bounds on the free variables. What effect this has on the results will be explained below. The right plot shows the relative volume that is remaining after each branch and bound step. This number should be very small, as that indicates that a lot of the initial search domain is removed from evaluation. With both runs ending up at a final ratio of less than 10^{-10} a lot of the initial search domain is lost. For the point solutions the the remaining relative box volume was always less than 10^{-13}. Continuing splitting to the tighter 0.01° box width criterion certainly helped getting to this lower value, but the fact that the point solutions are a four dimensional problem compared with a five dimensional problem for the continuous solutions cannot be overlooked when comparing these values. The lower value of the point solutions suggest that these results are more accurate than the continuous set. The elaboration below confirms this.

The target of trimming methods is to find the states where the accelerations are zero. The human vestibular system was found to be able to detect accelerations of $0.02m/s^2$ for translational motions and $0.05^\circ/s^2$ for rotational motions [25]. As these values are the limits of what humans can possibly detect they were not used as hard criteria. They can now be used as an indication for the quality of the results. For each solution the remaining accelerations are known. The hull of these accelerations is taken and its bounds are plotted in Figures 9 to 12. The most striking thing of these plots is that irrespective of the run the accelerations are much wider at high speeds compared to low speeds. The force that a control effector can exert on the aircraft body is a function of velocity. All the boxes are equally wide throughout the range of Mach numbers. That means that at low speeds a certain interval is multiplied with a small number, while at high speeds that same interval is multiplied with a much larger number. This results in wider intervals in terms of forces, and this caries through to the final accelerations. With this in mind it is not surprising that the only two trim points that meet the $0.02m/s^2$ and $0.05^\circ/s^2$ criteria are at 20,000 ft at the lowest two Mach numbers, 0.3000 and 0.4328. Because the available control forces also are a function of atmospheric density the widths of the accelerations are larger at lower altitudes. This lead to the best trim point at 10,000 ft being at Mach 0.3000. This solution only has two boxes out of 37 boxes violating the sensible acceleration limits. Comparing the point solutions to the set solution it becomes clear what effect the relaxed precision criteria has had: a factor ten less accurate in box accuracy leads to accelerations that are

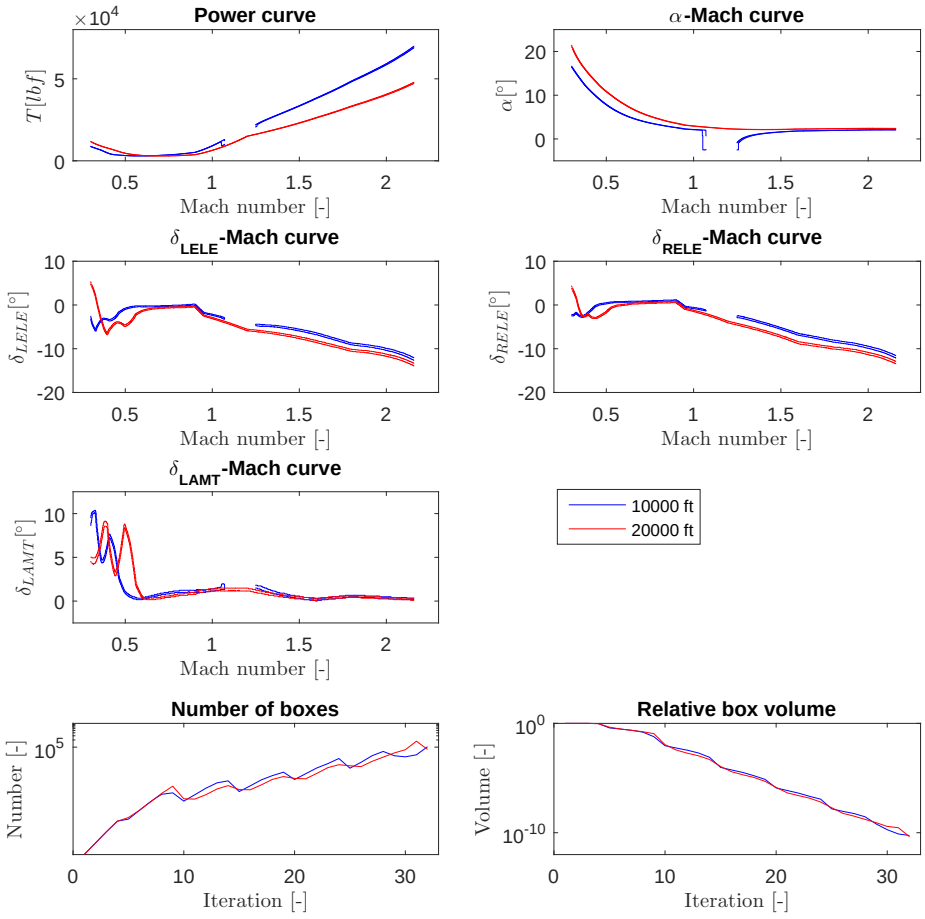

Fig. 8 Full interval analysis solution at 10000 ft and at 20000 ft

approximately a factor ten larger. The translational accelerations in x- and y-direction are acceptable, but the ones in z-direction are too large. In the full set solution rotational accelerations can be as large as $10°/s^2$. With this acceleration it only takes 8.49 seconds to do a full rotation. This is clearly not acceptable for a trim set. Further refinement is required, especially at high speeds. Keep in mind that every box does at least contain zero in the acceleration, otherwise it would have been removed from the list of boxes. So despite that the bounds on the control deflections less than $1°$ apart the bounds on the accelerations are too wide to accept these results as trim sets. The `fmincon` algorithm used in the trim routine that was supplied with the ICE model can have severe rotational accelerations at trim solutions. In one case these are as large as $34°/s^2$.

VI. Conclusion

Looking at the results it can be seen that interval analysis is a method that has great potential as a trimming method. With the current implementation it was possible to get bounds of a continuous trim profile for the highly overactuated ICE model that are only $1°$ wide. Within these $1°$ bounds the accelerations are still up to maxima $0.5m/s^2$ in translational motions and $10°/s^2$ in rotational motions. This is something that needs improvements before the trim sets can be accepted. The first potential for improved results is to make the model also calculate directional derivatives. These

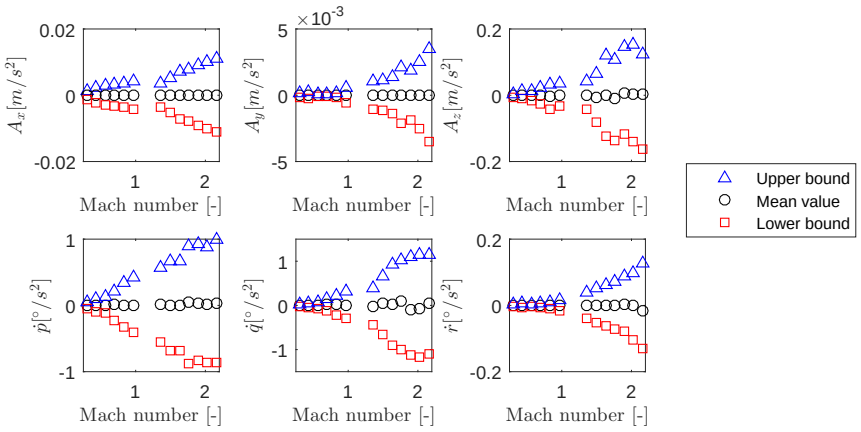

Fig. 9 Mean acceleration and bounds for trim points at 10000 ft

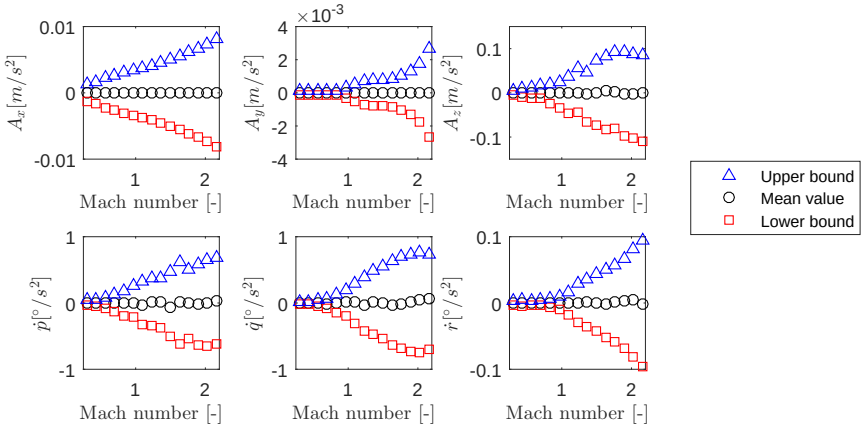

Fig. 10 Mean acceleration and bounds for trim points at 20000 ft

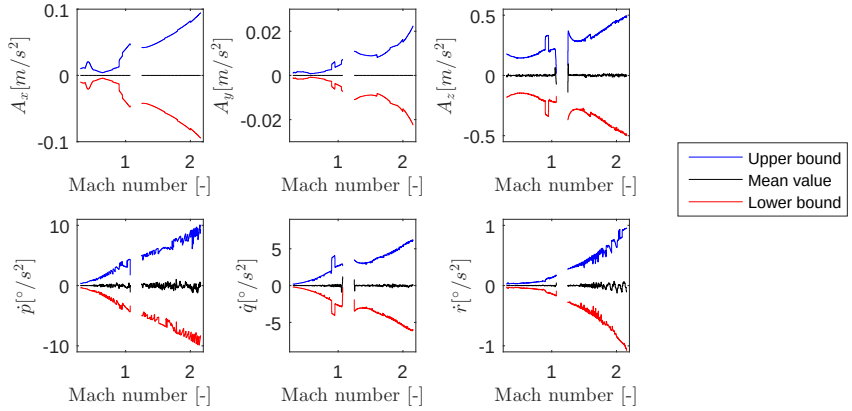

Fig. 11 Mean acceleration and bounds for trim set at 10000 ft

could then be used to implement the very efficient Newton interval algorithm. If there are no zeros in the derivative of a function the interval Newton algorithm has a huge box size reduction potential. The directional derivatives could

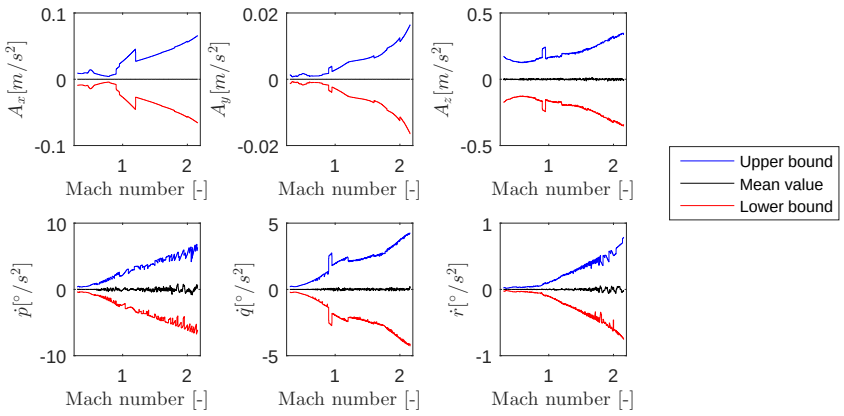

Fig. 12 Mean acceleration and bounds for trim set at 20000 ft

also be used to choose the splitting direction for the box splitting procedure. If the gradient is large in one direction it can be expected that it will result in more boxes that do not contain zeros, which keeps the boxcount low. The next improvement is aimed at the current implementations efficiency. In its current form the model can only evaluate one box at a time. With an average throughput of 160 boxes per minute this becomes impractical if high precision is desired. Because the code is written in Matlab, it should be possible to exploit its power to work with vectorized lists of boxes, which allows for all boxes to be calculated at once, which can allow for huge gains in computation times. The challenge that is not overcome for this work is rooted in the interval interpolation functions. Finally a multivariate B-spline model has been developed at the institution where this work was performed. If the interval routine was altered so that it could work with that model the whole interpolation of data tables would be made redundant, because spline models are fully analytical. Another advantage is that the directional derivatives can be directly calculated from the spline model as well. This indicates that there is a lot to gain from applying interval analysis to spline models. There is also a big downside to spline models. Spline functions are summations of basis functions and each basis function has one entry for the barycentric coordinates of the free variables. This means that interval arguments occur many times in a spline function. This can give rise to a big dependency problem. With the above improvements it might be possible to calculate trim sets with more free control effectors. The increase in calculation time after going from a four variable problem to a five variable problem was immense, and with the current implementation this is expected to happen again it a six variable problem is attempted. Furthermore the required number of boxes is expected to cause problems with memory limitations of current computers. Compared to the more conventional trim algorithm based that uses the Matlab function `fmincon` the found results are slightly different. Because the conventional method tries to minimize the squared sum of all control effectors it tends to find solutions that have smaller control deflections. The interval analysis trim procedure focuses purely at the accelerations of the model and tries to bound the zeros. This gives smaller leftover accelerations when very small box diameters are reached. It does require more time than the `fmincon` algorithm. The biggest drawback of the `fmincon` method is that the initial guess can influence whether or not a solution is found. Interval analysis does not have this issue. Instead the only input needed for it to start is the search domain. And despite the fact that it cannot find exact solutions like `fmincon` can it does have the ability to bound continuous trim sets over large domains. If the efficiency of the current approach can be increased the advantages of interval analysis can make it a trimming method that is very suitable for highly overactuated aircraft such as ICE with clear advantages over conventional trimming methods.

References

[1] ICAO, "A Coordinated, Risk-based Approach to Improving Global Aviation Safety," Tech. rep., International Civil Aviation Organization, Montreal, 2016.

[2] Kwatny, H. G., Dongmo, J.-E. T., Chang, B.-C., Bajpai, G., Yasar, M., and Belcastro, C., "Nonlinear Analysis of Aircraft Loss of Control," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 1, 2013, pp. 149, 162. doi:10.2514/1.56948.

[3] van Oort, E. R., Chu, Q. P., and Mulder, J. A., "Maneuver Envelope Determination through Reachability Analysis," *Advances in*

Aerospace Guidance, Navigation and Control, edited by F. Holzapfel and S. Theil, Springer-Verlag, Berlin Heidelberg, 2011, pp. 91–102.

[4] Dorsett, K. M., and Mehl, D. R., "Innovative Control Effectors (ICE)," Tech. rep., Wright-Patterson Air Force Base, Fort Worth, Texas, 1996.

[5] Hansen, E., and Walster, G. W., *Global Optimization Second Edition , Revised and Expanded*, Marcel Dekker, Inc, New York, 2004.

[6] Jaulin, L., Kieffer, M., Didrit, O., and Walter, E., *Applied Interval Analysis*, Springer-Verlag, 2001.

[7] van Kampen, E.-J., Chu, Q., Mulder, J., and van Emden, M., "Nonlinear Aircraft Trim Using Interval Analysis," *AIAA Guidance, Navigation and Control Conference and Exhibit*, , No. August, 2007. doi:10.2514/6.2007-6766.

[8] De Marco, a., Duke, E. L., and Berndt, J. S., "A General Solution to the Aircraft Trim Problem," *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*, , No. AUGUST 2007, 2007, pp. 1–40. doi:doi:10.2514/6.2007-6703.

[9] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006. doi:10.1137/S1052623499350013.

[10] Paranjape, A., Sinha, N. K., and Ananthkrishnan, N., "Use of Bifurcation and Continuation Methods for Aircraft Trim and Stability Analysis – A State-of the-Art," *Journal of Aerospace Sciences and Technologies*, Vol. 60, No. 2, 2008, pp. 85–100.

[11] Shankar, P., "Characterization of Aircraft Trim Points Using Continuation Methods and Bifurcation Analysis," *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013, pp. 1–12. doi:10.2514/6.2013-4924.

[12] Tupy, J., and Zelinka, I., "Evolutionary algorithms in aircraft trim optimization," *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*, 2008, pp. 524–530. doi:10.1109/DEXA.2008.98.

[13] Elgersma, M. R., and Morton, B. G., "Nonlinear Six-Degree-of-Freedom Aircraft Trim," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 2, 2000, pp. 305–311. doi:10.2514/2.4523.

[14] Moore, R. E., and Yang, C. T., "Interval Analysis 1," Tech. rep., Lockheed Aircraft Corporation Missiles and Space Division, Sunnyvale, California, 1959.

[15] Moore, E. R., Kearfott, R. B., and Cloud, M. J., *Global Optimization using Interval Analysis: Interval Optimization for Aerospace Applications*, Society for Industrial and Applied Mathematics, 2009.

[16] IEEE Standards Board, "IEEE Standard for Binary Floating-Point Arithmetic," , 1985.

[17] Van Kampen, E., *Global Optimization using Interval Analysis: Interval Optimization for Aerospace Applications*, Delft University of Technology, 2010.

[18] Juliana, S., *Re-entry Flight Clearance*, Delft University of Technology, 2006.

[19] Rump, S., "INTLAB - INTerval LABoratory," *Developments in Reliable Computing*, edited by T. Csendes, Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104. URL `http://www.ti3.tuhh.de/rump/`.

[20] Dorsett, K. M., and Mehl, D. R., "Innovative Control Effectors (ICE) Phase II," Tech. rep., Wright-Patterson Air Force Base, Fort Worth, Texas, 1997. doi:WL-TR-96-3043.

[21] Addington, G. A., and Myatt, J. H., "Control-Surface Deflection Effects on the Innovative Control Effectors (ICE 101) Design," Tech. Rep. Ice 101, Wright-Patterson Air Force Base, 2000.

[22] Neumaier, A., "Taylor Forms - Use and Limits," *Reliable Computing*, Vol. 9, No. February, 2003, pp. 43–79.

[23] Kraemer, W., "Generalized Intervals and the Dependency Problem," *Proceedings in Applied Mathematics and Mechanics*, Vol. 684, 2006, pp. 683–684. doi:10.1002/pamm.200610.

[24] Van Hentenryck, P., McAllester, D., and Kapur, D., "Solving Polynomial Systems Using a Branch and Prune Approach," *SIAM Journal on Numerical Analysis*, Vol. 34, No. 2, 1997, pp. 797–827.

[25] Heerspink, H. M., Berkouwer, W. R., Stroosma, O., and Mulder, M., "Evaluation of Vestibular Thresholds for Motion Detection in the SIMONA Research Simulator," *AIAA Modeling and Simulation Technologies Conference and Exhibit*, , No. August, 2005, pp. 1–21. doi:10.2514/6.2005-6502.

Chapter 2

Innovative Control Effectors model

This chapter will provide an overview and background of the ICE model. The ICE model will be the main object during this research, as the goal is to find its trim set. The first section will discuss the background information about the model. This includes an explanation why this model is particularly interesting for this research. The next section will present how the mathematical model is implemented.

2-1 Innovative Control Effectors Background

Application of new low-signature technologies to fighter aircraft puts more and tighter constraints on the design of such aircraft (Dorsett, Fears, & Houlden, 1996). To ensure a low radar cross section control surface edges should be aligned with other external edges, while vertical control surfaces should be omitted altogether. Furthermore development of multi-axis thrust vectoring has raised the bar in terms of maneuverability. Combining the two aims of reducing aircraft signature while being highly maneuverable has pushed the design envelope to new domains. Lockheed Martin developed the ICE model to meet these requirements. The main purpose of the ICE project was "*to investigate the potential for new and innovative methods for stabilization and control for high performance, low all-aspect signature fighters*" (Dorsett et al., 1996). The result of this investigation is a fighter aircraft model with two configurations. The 101-series configuration is a tailless delta wing with a 65° leading edge sweep angle developed for the air force. The 201-series configuration is a tailless canard-delta plan with a 42° leading edge sweep angle. This configuration has been developed for the navy and is also to be used on aircraft carriers. Both configurations are depicted in Figures 2-1 and 2-2.

The lack of a vertical tail means that different means of providing a yaw moment must be used. It has been shown that yaw trust vectoring is a very powerful way to do that at low speeds. At moderate to high speeds this yaw vectoring runs into engine and structural limitations, so other solutions must be implemented. A conceptual study compared five different concepts for lateral-directional controls, of which three are implemented in the ICE model. These

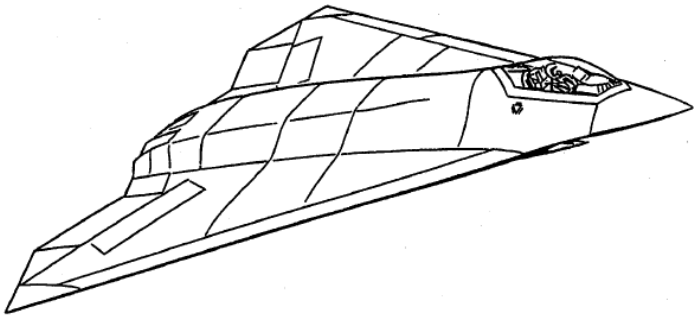

Figure 2-1: Impression of what the 101-series configuration looks like (Dorsett et al., 1996)

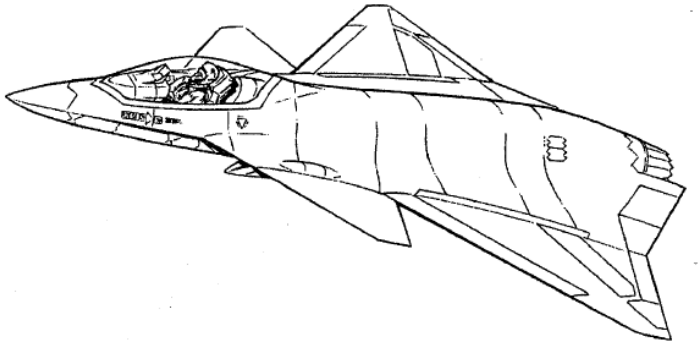

Figure 2-2: Impression of what the 201-series configuration looks like (Dorsett et al., 1996)

are all-moving wing tips (AMTs), spoiler-slot-deflectors (SSDs), and differential leading edge flaps (DLEFs). An impression of each of these three concepts is given in Figure 2-3. When combined with elevons, pitch flap (PF) and pitch thrust vectoring (PTV) and yaw thrust vectoring (YTV) as done on the ICE model the final model has a grand total of thirteen control effectors. Since there are multiple effectors that have the same purpose there can be redundancy in control power, which means that there could be several control effector deflection combinations that will result in the same flight conditions. This makes the ICE model a very interesting model for this research that aims to find all possible trim points of an aircraft model.

In (Dorsett et al., 1996) and (Dorsett et al., 1997) initial analysis and wind tunnel test results are given for both aircraft configurations. The most important conclusions are that the AMTs are the most effective control effector for full flight envelope operations. Up to 40° angle of attack operation can be done without significant decrease in control power. Going to higher angles of attack will see a drop in effectiveness but the yaw power will still be sufficient. Angle of attack smaller than 10° see an adverse rolling effect from the AMTs, while this effect becomes favorable at higher angles. The SSDs yaw power decreases as the angle of attack becomes bigger, stopping to be effective at approximately 30°. In combination with multi-axis thrust vectoring this problem can be overcome. The DLEFs are only effective at higher angles of attack and therefore in themself not very useful, but they can help to extend the envelope of the other solutions. Using both DLEFs and SSDs gives for example a favorable aerodynamic interaction at high angle of attack. There are more interactions going

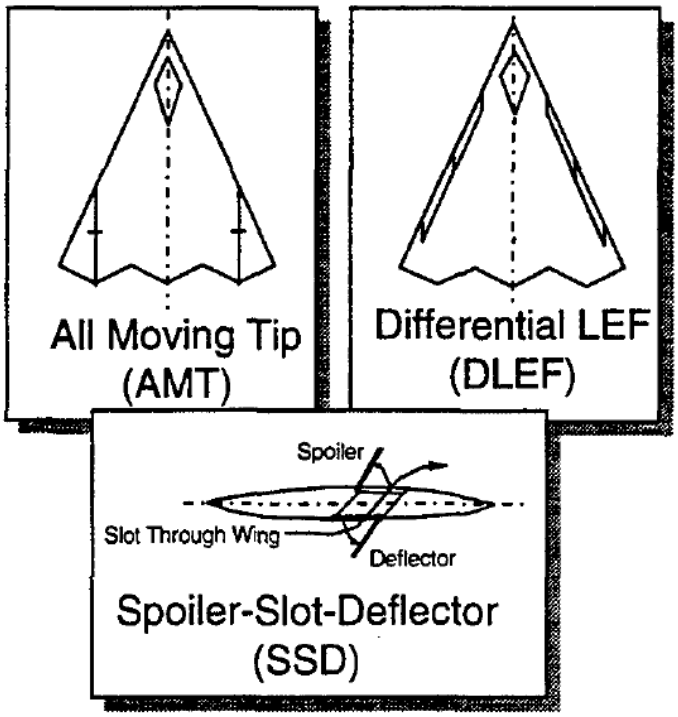

Figure 2-3: Concepts for yaw moment generation (Dorsett et al., 1997)

on. Having a leading edge deflection creates more adverse rolling dynamics from the AMTs. AMTs deflections also degrade the performance of the adjacent trailing edge device, in this case the elevons. At high angle of attack using the SSDs improved the functionality of the pitch flap. Table 2-1 shows the physical limitations of the control effectors. In Table 2-2 the sign conventions that will be used in the rest of this research are mentioned.

2-2 Mathematical model

The flight dynamics of the 101-configuration of the ICE model described in the previous section are available in a Simulink model provided by Lockheed Martin. As the 201-configuration

Table 2-1: Control effector limitations for ICE (Dorsett et al., 1996)

Control effector	Lower limit [deg]	Upper limit [deg]	Rate limit [deg/s]
AMT	0	60	150
DLEF inboard	0	40	40
DLEF outboard	-40	40	40
Elevons	-30	30	150
PF	-30	30	50
SSD	0	60	150
PTV	-15	15	60
YTV	-15	15	60

Table 2-2: Control effector sign conventions (Dorsett et al., 1996)

Control effector	Sign convention
AMT	Trailing edge down
DLEF inboard	Leading edge down
DLEF outboard	Leading edge down
Elevons	Trailing edge down
PF	Trailing edge down
SSD	Trailing edge up
PTV	Nozzle end down
YTV	Nozzle end left

is not available, this research will focus solely on the 101-configuration. The Simulink implementation of the airframe model is shown in Figure 2-4. Along with the model came a Matlab script that can determine a trim set. On a higher level the trim values are added to the user supplied control inputs before they are fed into the airframe. With these inputs the airframe calculates all the states and flight parameters of the aircraft.

In the 'Aerodynamic Coefficients' block the aircraft's force and moment coefficients are calculated. These come from equations that in turn each depend on 17 to 19 other coefficients. The values for these coefficients are stored in 108 data tables that are constructed based on wind tunnel test results. The equations can be found in Appendix A. As an alternative to this look-up table structure, a spline model is available. Using spline models can be advantageous since they have a defined value everywhere in the domain. This removes the need for interpolation and extrapolation during calculations on the aircraft.

2-3 Conclusion

This chapter has given an overview of the ICE model. The background of the project has been mentioned as well as some important characteristics. These include three innovative concepts to provide yaw power. The implementation of model has been shown and explained as well.

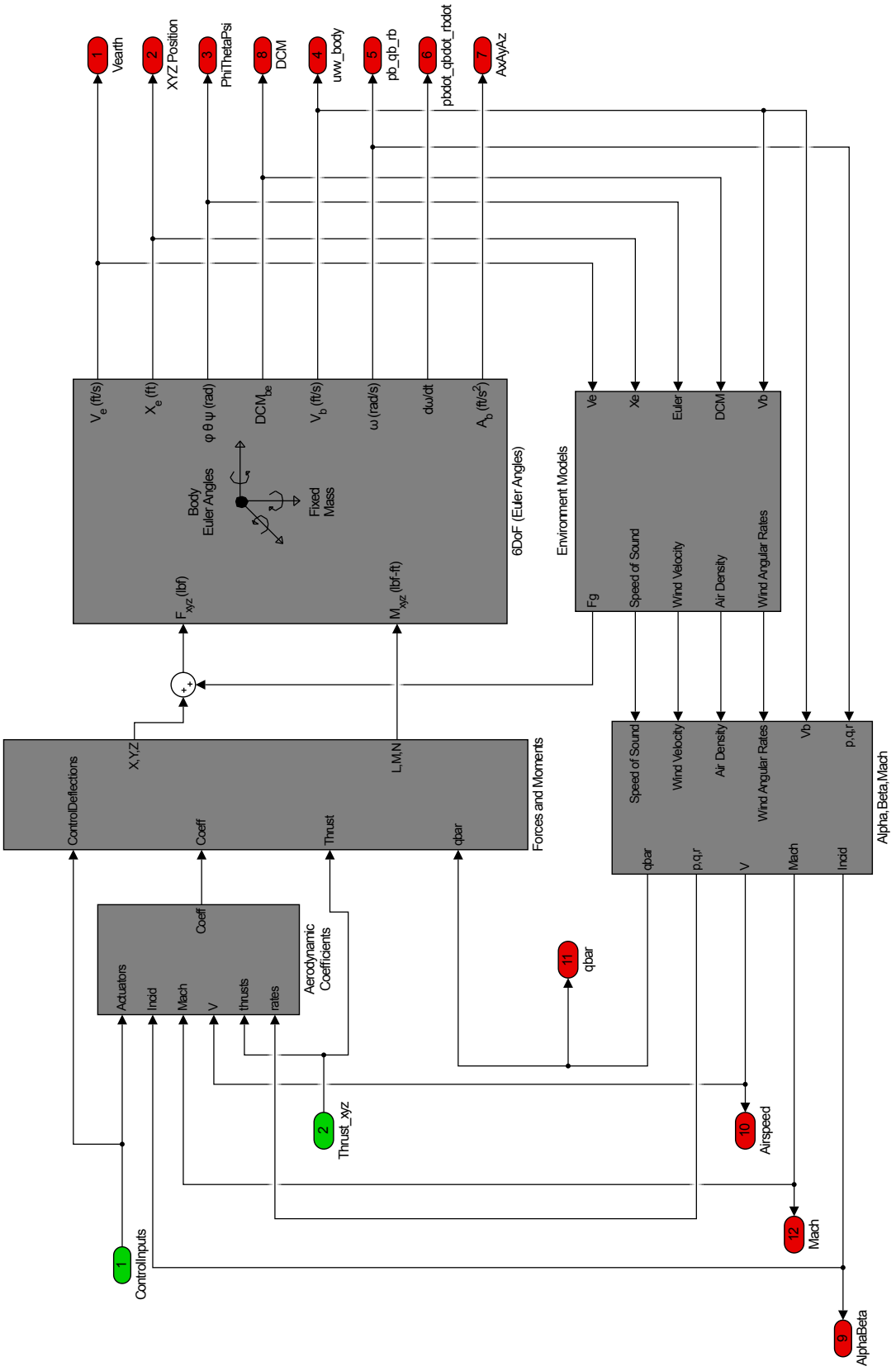

Figure 2-4: Implementation of ICE as provided by Lockheed Martin

Chapter 3

The trim problem

The research done in this work is mainly concerned with finding the full trim set of the ICE aircraft model. This chapter will start with discussing what trim points are. The next section will give a short overview of methods that can be used for finding these trim points. This is referred to as solving the trim problem. Both the strong and weak points of each method will be discussed, including information about computational load, running time and convergence rate if available. The chapter ends with an explanation if and why interval analysis would be the most suitable method to find the trim points of the ICE model.

3-1 Trim points

Finding trim points is a vital task for many engineering studies involving aircraft as there are many applications for them. Trim points can be defined as the steady states of the aircraft, so the aircraft is in an equilibrium condition. When expressing an aircraft as a system of equations in the shape of $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ the mathematical definition of equilibrium is

$$f(\mathbf{x}, \mathbf{u}) = 0 \tag{3-1}$$

$$\mathbf{x} = [u\ v\ w\ p\ q\ r\ N\ E\ D\ \phi\ \theta\ \psi]^T \tag{3-2}$$

In these equations $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the input vector. Because none of the states are changing in a trim point the aircraft will simply keep doing what it is doing, which would make for a safe flight condition. Consider now the aircraft state vector in Equation 3-2. The first six elements of this state vector represent velocities, the next three are positions while the last three are orientations. When taking the time derivative of these components the result will be six accelerations and six velocities. If the definition in Equation 3-1 would strictly be applied there would be no velocities allowed and the aircraft could not move. The only way this could be satisfied is when the aircraft is standing on the runway. Fortunately

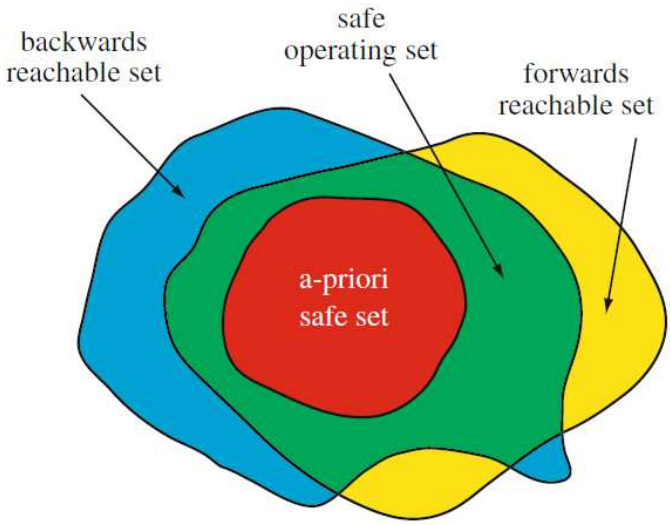

Figure 3-1: Determining the safe flight envelope based on the safe trim set (Van Oort et al., 2011)

Newton's laws define equilibrium as a state in which all forces and moments cancel each other out so that there are no accelerations. This dynamic equilibrium is what one should look for when looking for trim points. Because in such conditions the aircraft will keep doing whatever it is doing it can be considered a safe flight condition. This feature can be used in the determination of the safe flight envelope of an aircraft. In (Van Oort et al., 2011) the trim set is taken as an initial set of safe flight conditions. All states from which this safe set can be reached is called the backwards reachable set. Contrary, all the states that the aircraft can maneuver to from the safe set is called the forwards reachable set. The intersection of the backwards and forwards reachable set gives the safe flight envelope. This is shown graphically in Figure 3-1.

Furthermore trim points are the points around which linear models are derived, as well as the design points for autopilots (De Marco et al., 2007). In flight simulation trim conditions are also used. Here they provide flight conditions that the simulation can be initialized on. This ensures that the simulation is closely matched to the real aircraft. Trim points are also a great help when comparing different flight models with each other, or when comparing a model to test flight data. As finding the trim points is not at all a trivial task there have been a lot of different approaches to solving the trim problem. All these methods attempt to find the zeros of the equations of motion, as is expressed in Equation 3-1. Some methods solve directly on the system, while others use a cost function.

3-2 Trimming methods

This section will discuss some potential trimming methods. Their strong and weak points are mentioned, as well as how they have been applied in the past.

3-2-1 Gradient-based method

If the trim problem is formulated in a cost function gradient-based solvers are a simple means to find the minimum of this function. In (De Marco et al., 2007) a good option for a cost function is given. How it is defined is shown in Equation 3-3.

$$J = \mathbf{x}^T \cdot \mathbf{W} \cdot \mathbf{x} \tag{3-3}$$

In this equation $\mathbf{x}$ is the vector of states that need to be equal to zero and $\mathbf{W}$ is a weighting matrix. It is assumed that this cost function is smooth and that its gradient ∇J is available or can be computed. By starting at a random point the direction of steepest decent is determined. In this direction a line search for the minimum is done. At this minimum value a new direction is established until ∇J is zero. This method has no guarantee that it will not get stuck in a local minimum, nor that it will find all possible solutions even if a lot of random start points are used. It is also noted that sometimes derivatives of aerodynamic coefficients or propulsion forces are needed. For some models these might not be available and require numerical reconstruction, which can be undesirable.

3-2-2 Closed form method

In (Elgersma & Morton, 2000) a closed form formulation for the equations of motion (EoM) of a nonlinear six degree-of-freedom aircraft model is used to calculate the trim points. The aircraft model itself contains gravity contributions, nonlinear angular rates and tabular aerodynamic functions. The model is linear in the four control inputs thrust, aileron deflection, elevator deflection and rudder deflection. This allows the EoM to be solved analytically, hence all possible trim conditions will be found. However the linearity in input means that varying the controls can only affect a four-dimensional subset of the full six-dimensional state space for a given state. During the solving for the control surface deflections no actuator saturation is accounted for. This can make some solutions invalid.

3-2-3 Simplex method

Applying the simplex method is not obvious for trim point determination, as it only works on linear problems. That it was possible to use it was due to a specific application that used active aeroelastic wings (Zink, Mavris, & Raveh, 2000). By using wing twist as a means to generate a moment on the aircraft body the conventional control effectors become more like trim tabs. In symmetrical maneuvers the trimming problem at hand is to determine the wing control surface deflections to tailor the load distribution over the wing to reduce the wing root bending, ultimately reducing structural weight. Trimming for asymmetrical maneuvers focuses on minimizing the hinge moments on the control deflectors. Because this trim problem actually tries to solve structural equations that can legitimately be written as a linear programming problem the simplex method can be applied and the results are valid. But when this method will be used to solve the EoM results will not be useful because generally EoM are highly nonlinear equations.

3-2-4 Sequential quadratic programming

A widely used optimization algorithm to solve nonlinearly constrained problems is sequential quadratic programming (SQP). As the name implies it achieves this by solving a sequence of quadratic programming subproblems. The constraints for these subproblems are linearizations of the original constraints, while its objective function is a quadratic approximation of the Lagrangian of the full problem (Gill, Murray, & Saunders, 2002). In (Schittkowski, 1986) a comparison between several solvers is made and it is shown that SQP programs work very effectively, faster and more accurately than the alternatives. It does need a starting point to start solving from, and this makes it hard if not impossible to ensure that all solutions are found. An example of an SQP algorithm is the Matlab function `trim`.

3-2-5 Bifurcation and continuation methods

In (Paranjape, Sinha, & Ananthkrishnan, 2008) a summary is given of bifurcation and continuation methods and how they have been applied. At an initial flight condition the aircraft model is trimmed. Some of the parameters of the aircraft are locked in the trimmed position. Of the remaining variables one is chosen to be the control parameter. By changing the control parameter the other free variables have to change as well to keep the aircraft trimmed. By recording these values a trim set is calculated. By determining the Jacobian and its eigenvalues for each trim configuration one can determine the stability of the system. A change in the stability is called a bifurcation.

A routine implemented in (Shankar, 2013) searches for the longitudinal trim set of a high fidelity F-16 model. At the start of each run an initial velocity is chosen. The aircraft model is trimmed for this velocity, giving settings for the elevator, leading edge deflectors and the thrust, as well as the required angle of attack. During each run the leading edge deflectors and thrust setting are kept the same, while the steady state velocity and angle of attack are calculated as a result of changing the elevator angle. This gives a trim set for each different setting of thrust and leading edge deflectors. An example set as can be found with continuation methods is shown in Figure 3-2. To get the full trim set a lot of runs would be required, even for a limited set of control effectors such as the ones used in this case.

3-2-6 Evolutionary algorithms

As shown by (Tupy & Zelinka, 2008), evolutionary algorithms have potential for finding trim conditions of aircraft. The parameters of the cost function in the shape of $J = \mathbf{W} \cdot |\mathbf{x}|$ are changed in a way similar to the process of evolution. Part of the parameters are randomly changed, if this gives a better solution there is a bigger chance that they will be retained. This process is repeated for a predetermined number of generations, where each generation is basically an iteration of the algorithm. A very strong capability of this method is that the nonlinear aircraft model is fully embedded in the trim routine without simplification. On the other hand it was realized that the method as of yet is not fast nor reliable enough to generate conclusive final results. However, the results are sufficient to do initial calculations to evaluate the effect of changing governing parameters withing the routine itself. As a final remark it is noted that the element of coincidence of evolutionary algorithms still is a weak

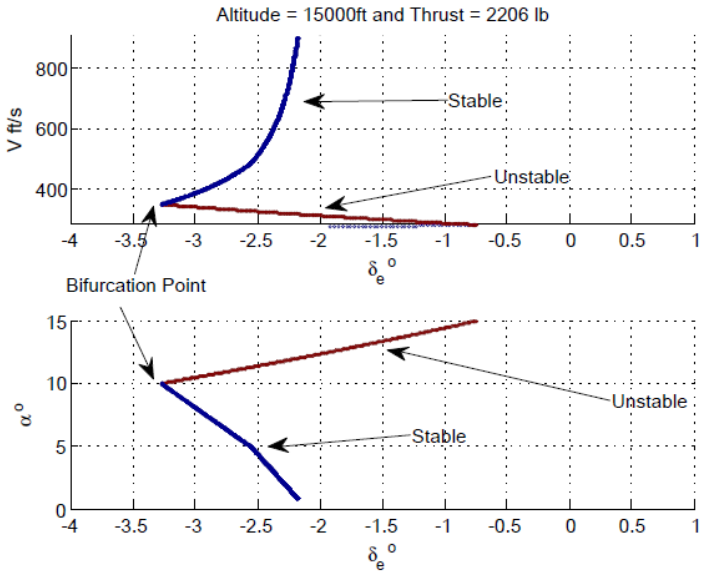

Figure 3-2: An example result from a continuation analysis showing a family of trim points (Shankar, 2013)

point that needs improving before this method could be more widely applied and accepted as a viable method for finding trim conditions.

3-2-7 Interval Analysis

In (Van Kampen, Chu, Mulder, & van Emden, 2007) an interval routine is proposed as a new approach to the trim problem. It utilizes the strong features of interval analysis to find all the trim points of an F-16 model in level and straight flight. The algorithm is demonstrated to be capable of finding all steady states in the search domain for a specified airspeed and altitude. This is illustrated in Figure 3-3, were the results found with interval analysis are compared to those found with sequential quadratic programming. It can be seen there that interval analysis finds both possible angles of attack, while SQP only finds one for each throttle setting. Interval analysis works by evaluating the system response over the entire domain, which is a set of intervals. The responses are also expressed as intervals. By using box contraction methods the search domain can be reduced. When no stopping criterion are met the search box gets split into smaller boxes, for which the process is repeated. This goes on until the value of the responses meets a certain value or the box dimensions meet the desired accuracy. The final list of boxes will contain the trim points. In order to get the responses of the aircraft model data tables must be interpolated in intervals. How this can be done linearly is shown in Figure 3-4. The value of C_{m0} must be evaluated at all the crosses in this figure. Because the interpolation is done linearly taking the minimum and maximum value from these points ensures the tightest possible interval from the data tables. The fact that every parameter can be expressed as an interval makes it also very easy to apply constraints to a model. By constraining the aircraft to go to a certain flight condition the box simply gets smaller. It is also possible to reduce the dimension of the entire problem by locking a free parameter, for example rudder deflection, to a crisp number. As a result the algorithm

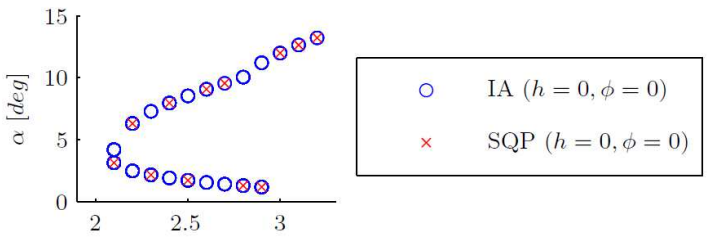

Figure 3-3: Example solutions for angle of attack found with interval analysis and sequential quadratic programming (Van Kampen, 2010)

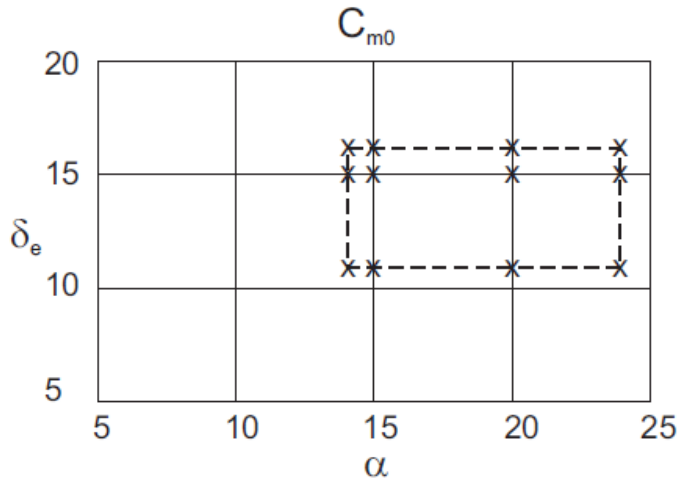

Figure 3-4: All points that need to be considered during linear interpolation in an interval box (Van Kampen et al., 2007)

does not need to solve for rudder deflection, but will only find all trimmed states with this fixed rudder deflection. This can be a very helpful feature, as interval analysis optimization is computationally quite expensive.

3-3 Conclusion

In this chapter a definition for trim is given. The shortest way to describe the trim set is that it contains all the dynamic equilibrium conditions. Several ways to find trim points have been discussed. Of these methods interval analysis has the most potential for finding the full trim set of the ICE model, because it is certain to find all trim points without fail. The amount of control effectors lead to the need of additional constraints on the problem, and interval analysis is very good in dealing with those. This makes it an extremely powerful trim method, despite it being computationally expensive. The next chapter will elaborate more on interval analysis in general and on interval analysis as an optimization method.

Chapter 4

Interval analysis

Interval analysis is a part of mathematics that works on interval numbers instead of crisp numbers. What this means is that every number spans a certain width, with an upper and a lower bound. Initially developed as a tool to quantify mathematical rounding errors of all kinds (R. E. Moore & Yang, 1959), it was soon realized that interval analysis could also be used as an optimization method (E. R. Moore, Kearfott, & Cloud, 2009). When used as an optimization method it is guaranteed that the global optimum is found and bounds are given for the location and value of the minimum (Walster, 2004). This feature has resulted in interval analysis being applied in a variety of ways, such as parameter estimation, robust control and robotics (Jaulin, Kieffer, Didrit, & Walter, 2001) or for nonlinear aircraft trim, optimization of human perception modeling and spacecraft rendezvouz and docking (Van Kampen, 2010). It has also been used to evaluate the effect of uncertainties for re-entry flight clearance (Juliana, 2006).

This chapter will give a short introduction to interval analysis as an optimization tool. The first section will define how the notation of intervals is done in this work, as well as some fundamental mathematical operators. Next the properties of interval functions will be discussed. This is followed by a description of a baseline interval analysis routine to solve optimization problems. With some minor alterations the optimization algorithm can be changed to a zero finding program. It will be explained how that is especially useful for aircraft trim. The next section will explain methods to enhance the interval analysis algorithm to improve convergence rate and accuracy. After the Matlab interval toolbox Intlab is described a very brief introduction to using interval analysis with spline models is given as well.

4-1 Interval notation and operators

In order to make the interval calculations in this work clear some definitions and conventions on notation are in order. The notation for crisp numbers has the convention of x for scalars, $\mathbf{x}$ for vectors and $\mathbf{X}$ for matrices. For intervals the same approach will hold, with the addition of square brackets around the symbols. So $[x]$ for an interval number, $[\mathbf{x}]$ for an interval

vector and $[\mathbf{X}]$ for an interval matrix. With the notation in place a clear definition of what an interval number or simply an interval can be made. The interval $[x] = [a, b]$ consists of the entire set $\{x^*|a \leq x^* \leq b\}$, that is all the crisp numbers x^* between and including a and b. x^* can represent any crisp number included in the interval $[x]$. The lower boundary of $[x]$, also known as the infimum, is denoted as $\underline{[x]}$ and is equal to a. Its upper boundary, the supremum, is denoted as $\overline{[x]}$ and is equal to b. For these values there are specific operators, as well as for the width and the midpoint of an interval. These are listed below.

$$inf([x]) = a \tag{4-1}$$

$$sup([x]) = b \tag{4-2}$$

$$w([x]) = sup([x]) - inf([x]) \tag{4-3}$$

$$mid([x]) = \frac{inf([x]) + sup([x])}{2} \tag{4-4}$$

Basic mathematical operators work in a specific way with intervals as shown in (Walster, 2004) and (Van Kampen, 2010). In the following consider the intervals $[x] = [a, b]$ and $[y] = [c, d]$. If any operator is expressed as $\bullet$ then

$$[x] \bullet [y] = \{x \bullet y | x \in [x], y \in [y]\} \tag{4-5}$$

This means that the interval $[x] \bullet [y]$ contains every possible number that results from any combination $x \in [x]$ and $y \in [y]$ with the specific operator. For addition this becomes

$$[x] + [y] = [a + c, b + d] \tag{4-6}$$

Subtraction follows the same principle.

$$[x] - [y] = [a - d, b - c] \tag{4-7}$$

While addition and subtraction are relatively easy in that they always require the same combination of upper and lower values, multiplication is slightly more difficult.

$$[x] \cdot [y] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)] \tag{4-8}$$

Dividing by zero is not allowed in crisp arithmatics. Therefore intervals containing zero are excluded for now. If one inverts one of the intervals as shown in Equation 4-9 division will follow the same rules as multiplication.

$$\frac{1}{[y]} = \left[\frac{1}{d}, \frac{1}{c}\right] \tag{4-9}$$

Table 4-1: Example showing interval analysis operations for $[x] = [-1, 3]$ and $[y] = [2, 7]$

Expression	Result
$[x] + [y]$	$[1, 10]$
$[x] - [y]$	$[-8, 1]$
$[x] \cdot [y]$	$[-7, 21]$
$\frac{[x]}{[y]}$	$[-0.5, 1.5]$

Table 4-2: Example interval analysis function for $[x] = [-1, 3]$ or $x = 2$

Expression	Interval results	Crisp results
$[x]^2 - [x]$	$[-3, 10]$	2
$[x]\,([x] - 1)$	$[-6, 6]$	2
$\left([x] - \frac{1}{2}\right) - \frac{1}{4}$	$[-0.25, 6]$	2

4-2 Interval functions

When using the basic arithmetic operations described in the previous section one will soon see some peculiarities of interval analysis. An example will be used to clarify this statement. Assume the intervals $[x] = [-1, 3]$ and $[y] = [2, 7]$. The result of the four basic operations is shown in Table 4-1. When performing operations with intervals it is important to keep in mind that addition and subtraction always causes the interval width to increase. The case for division shows that this is not always true for multiplication and division.

Now consider the three expressions in Table 4-2. If $[x] = [-1, 3]$ were used as an argument the resulting intervals would be different for each different expression. Each of the expressions has a different inclusion interval. If for example the crisp number $x = 2$ was used the result for each of the three different expressions is the same. This indicates that how a function is written determines the width of the resulting interval. What one should aim for is to always get the minimum inclusion interval. This interval is the narrowest interval that contains the function. Its lower boundary is the minimum value of the function on the domain, its upper boundary the maximum. A rule of thumb is to keep the number of occurrences of an interval in the function expression to a minimum.

The final cases that will receive extra attention are shown in Table 4-3. For the first expression it would be logical to expect the result of this operation to be $[0, 0]$. This is however not the case. The reason that this happens is known as dependence. What actually happens in the operation is $\{x - y | x \in [x], y \in [x]\}$. This shows that the numbers that are being subtracted are completely independent. Therefore there is no reason for them to be the same. The same happens in the second expression where a random number in $[x]$ gets multiplied with an other random number that is also in $[x]$. Mathematically, it actually states $\{x \cdot y | x \in [x], y \in [x]\}$. When taking the square of an interval number however, as in the final expression, the operation is $\left\{x^2 = x * x | x \in [x]\right\}$. This causes the interval to be tighter when squaring compared to when an interval is simply multiplied with itself. This phenomenon is known as interval dependence.

Table 4-3: Example showing dependence for interval analysis operations for $[x] = [-1, 3]$

Expression	Result
$[x] - [x]$	$[-4, 4]$
$[x] * [x]$	$[-3, 9]$
$[x]^2$	$[0, 9]$

4-3 Interval analysis optimization algorithm

In (Van Kampen, 2010) an interval analysis optimization routine is explained and applied. That routine is used here to explain how these type of algorithms work. A flowchart of the process is given in Figure 4-1. To start the domain is set. The vector of interval inputs becomes the first box. The cost function is evaluated over this box. This gives an interval that represents all values that the cost function can take within the box. The upper bound of this interval is set as the first estimate for the minimum of the problem. Both the box and the cost interval are added to the list of boxes. This concludes the initialization phase of the algorithm. The algorithm continues by taking the first box from the list. Box contraction algorithms are performed to see if the box can be reduced in size already. After these contractions the box and its result are compared to stopping criteria, which can be something like a certain box size. If they are met the box gets added to the list of solutions, if they are not the box gets split into smaller boxes. The cost function is again evaluated for all the new boxes. If the lower boundary of cost interval is higher than the current estimate of the minimum the box can be deleted as it is certain that the minimum is not contained in the box. By comparing the upper boundaries of the remaining boxes to the current estimate of the minimum a new estimate can be found. The lowest upper bound becomes the new estimate. The boxes are added to the list of boxes, after which the list is clipped based on the current estimate of the minimum. This procedure keeps looping through the boxes until all boxes are added to the list of solutions.

4-4 Using interval analysis for aircraft trim

In the method described above a check is done during each iteration to see if the estimate of the minimum needs updating. This ensures that the global minimum of a function is found. Instead of looking for an estimate of the minimum it can be interesting to look for all locations that a function assumes a certain value. This way interval analysis can be used for zero finding. To implement a zero finding algorithm only minor changes need to be made to the minimization algorithm. The comparison of the estimate of the minimum with the lower bound of each box now needs to be changed to a check to see if zero is contained by the box. It is known that this is not true if the upper bound is smaller than zero or if the lower bound is larger than zero. The step updating the estimate of the minimum can be removed altogether. This algorithm can be generalized to find locations that have different values than zero, and also to interval values. As explained in Chapter 3, trimming an aircraft involves finding the states where accelerations are zero. An aircraft model can be used to calculate the body forces and moments, and thus by Newton's second law accelerations. When using interval analysis, the input to a model is a box consisting of the ranges of the free variables. These are

flight parameters such as α and mach number combined with control deflections and thrust settings. By setting some of these to fixed values the dimensionality of the problem can be reduced, in order to reduce computation time. This is an effective way to use constraint handling capabilities of interval analysis. If there are no zeros in the model accelerations for a box than it is known that there cannot be a trim point in that box, so it does not require further evaluation and can be discarded. If there are zeros there could be trim points in that box, so it gets split and the subboxes are checked again for zeros. This process is repeated until all stopping criteria are met. The boxes that were removed because they reached the desired precision contain trim points. If the precision is high enough, the input boxes are very small. They should be so small that the upper bound and the lower bound hardly give a different model response. When that is the case a single value from the box can be used as a crisp input to the aircraft model. This shows how the zero finding capabilities of interval analysis can be used to find aircraft trim points.

Box contraction algorithms

In the description of an interval analysis optimization algorithm box contraction algorithms were mentioned. Here three of them will be explained. The first one is the interval Newton method, the second is box consistency and the third is hull consistency.

Newton's method

In (Van Kampen, 2010) the interval Newton method is explained. It is based on the mean value theorem and is a very powerful method of finding zeros of functions. According to the mean value theorem there exists a point ξ between the points x and x^* for any differentiable function f such that:

$$f'(\xi) = \frac{f(x) - f(x^*)}{x - x^*} \tag{4-10}$$

If in this case x^* is a zero of the function, then it can be derived form Equation 4-10 that:

$$x^* = x - \frac{f(x)}{f'(\xi)} \tag{4-11}$$

If both x and x^* are contained in the interval $[x]$ then ξ is in $[x]$ as well, since it is in between x and x^*. From this it also follows that:

$$x^* \in x - \frac{f(x)}{f'([x])} \tag{4-12}$$

and if

$$N(x, [x]) = x - \frac{f(x)}{f'([x])} \tag{4-13}$$

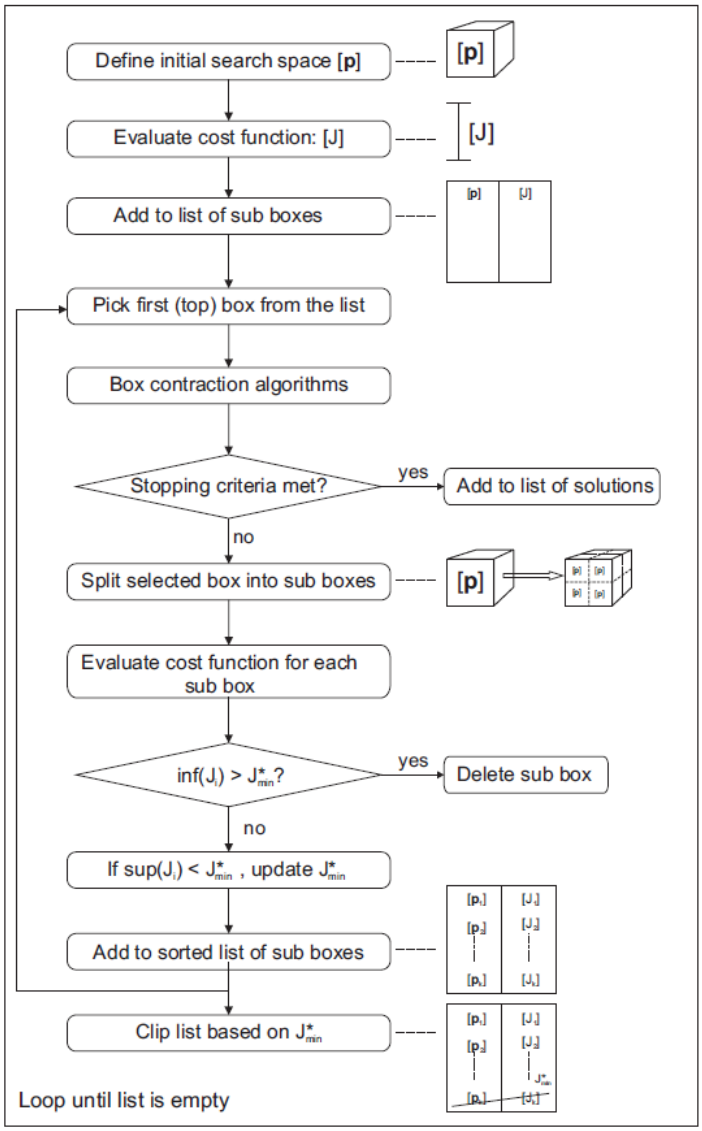

Figure 4-1: Flowchart showing the steps in an interval analysis optimization (Van Kampen, 2010)

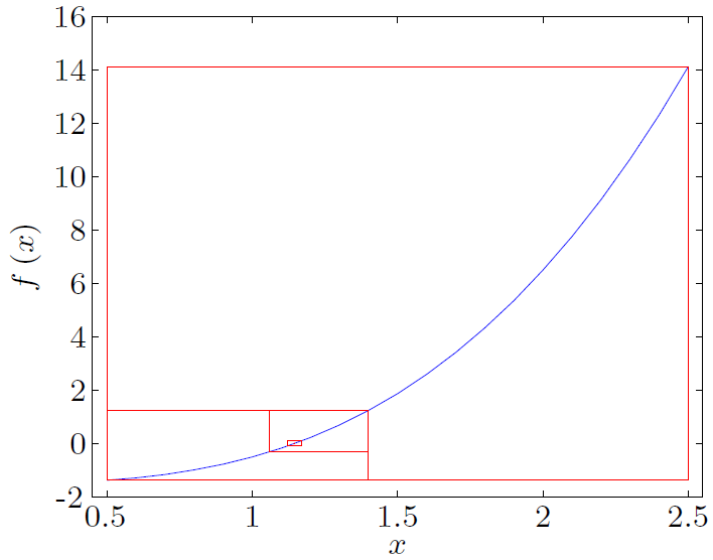

Figure 4-2: Example showing how the interval Newton method narrows around the zero of a function (Van Kampen, 2010)

it becomes obvious that $N(x,[x])$ will contain any zeros of f that are also contained in $[x]$. This in turn means that the zeros have to be in the intersection of $[x]$ and $N(x,[x])$. All of this can be put together in an algorithm that becomes more narrow around the zero x^* as the iterations go on, as is shown in Figure 4-2.

$$x_n \in [x_n]\ N(x_n,[x_n]) \quad = x_n - \frac{f(x_n)}{f'([x_n])}\ [x_{n+1}] = [x_n] \cap N(x_n,[x_n]) \tag{4-14}$$

Box consistency

Box consistency works by checking of parts of the domain of a box are inconsistent with the problems function. Here the algorithm derived in (Walster, 2004) will be explained. Suppose a nonlinear function:

$$f(x_1,...,x_n) = 0 \tag{4-15}$$

Box consistency tries to remove subboxes in $[\mathbf{x}] = ([x_1],...,[x_n])$ that do not give a solution to $f([\mathbf{x}]) = 0$. This is achieved by inserting the full interval for all variables but one, which leads to

$$q(x_i) = f([x_1],...,[x_{i-1}],x_i,[x_{i+1}],...,[x_n]) \tag{4-16}$$

If for x_i in the subinterval $[\overline{x}_i]$ of $[x_i]$ $q(x_i) \neq 0$ then it is known that $[\overline{x}_i]$ does not give consistent solutions for the function. This means that the subbox $([x_1],...,[x_{i-1}],[\overline{x}_i],[x_{i+1}],...,[x_n])$ can be discarded.

Hull consistency

The principle of hull consistency is using multiple equivalent expressions to reduce the box width. The explanation and example given here can be found in (Van Kampen, 2010) as well. Most functions can be split into two parts:

$$f(x) = g(x) - h(x) = 0 \tag{4-17}$$

The search domain is assumed to contain the solution interval $[x]$ but with the expression $g(x) - h(x)$ an alternative method for finding $[x]$ becomes available as well:

$$[x_{alt}] = g^{-1}\left(h\left([x]\right)\right) \tag{4-18}$$

Since both solutions are equally valid, taking the intersection of the two solutions generally lead to tighter intervals:

$$[x_{new}] = [x] \cap [x_{alt}] \tag{4-19}$$

Now for an example take the function shown in Equation 4-20 with the initial search domain of $[-100, 100]$.

$$f(x) = x^4 + x - 2 = 0 \tag{4-20}$$

This function can be splitted in two parts:

$$\begin{aligned} g(x) &= x^4 \\ h(x) &= 2 - x \end{aligned} \tag{4-21}$$

Inserting the search domain into $h(x)$ yields $[-98, 102]$. Using this combined with the fact that x^4 cannot be negative leads to:

$$g^{-1}\left(h\left([x]\right)\right) = [-98, 102]^{\frac{1}{4}} = [0, 102]^{\frac{1}{4}} = [-3.18, 3.18] \tag{4-22}$$

This can be used to find the smaller interval box that contains the solution.

$$[x_{new}] = [x] \cap g^{-1}\left(h\left([x]\right)\right) = [-100, 100] \cap [-3.18, 3.18] = [-3.18, 3.18] \tag{4-23}$$

This is an enormous improvement with respect to the initial box. In practice it is often quite difficult to know how to split the function and in which order. This limits the practicality of hull consistency methods.

4-5 Implementing interval analysis

For using interval analysis in Matlab the toolbox Intlab will be used (Rump, 1999). Intlab is designed to be almost as fast as regular floating point operations. It has also been designed very extensively, ensuring that anything Matlab can do can also be done with intervals if the Intlab toolbox is used. Intlab is completely written in Matlab, so should something not work for whatever reason it can be fixed. In (Hargreaves, 2002) the fast performance of Intlab is acknowledged by mentioning that the availability of such toolboxes has greatly increased the popularity of interval analysis. A short tutorial is also given to get to know how to work with Intlab quickly. Unfortunately the Intlab toolbox is not compatible with Simulink, so in order to make Intlab work with the ICE model some conversions must be made. These are discussed in Chapter 6.

4-6 Interval analysis and B-splines

As mentioned in Chapter 2 a B-splines model of the ICE model is available as an alternative to the data look-up table version. A spline model is a summation of local basis functions that are distributed over an underlying triangulation of simplices (De Visser, Chu, & Mulder, 2009). On each simplex the locations are evaluated based on the local barycentric coordinate system. In order to do calculations with splines the physical coordinates of a certain state must be converted into the local coordinates. A function can then be evaluated by summing the local basis functions scaled with the B-coefficients. The definition of a basis function is shown in Equation 4-24. In this equation d is the degree of the spline polynomial, κ the associated multi-index and b_i are the barycentric coordinates.

$$B_{\kappa}^{d}(b) = \frac{d!}{\kappa!} b^{\kappa} = \frac{d!}{\kappa!} b_0^{\kappa_0} b_1^{\kappa_1} ... b_n^{\kappa_n} \tag{4-24}$$

This shows a potential pitfall for B-spline models and interval analysis. A consequence of summing the basis functions is that each barycentric coordinate occurs a lot in each function expression. If these coordinates are governed by physical interval coordinates this means that there are a lot of interval occurrences in the function expressions. This causes dependency, so the intervals calculated by spline models can be much wider than desired. There are also advantages of spline models, in that the directional derivatives can be directly calculated from the same set of B-coefficients (De Visser, Chu, & Mulder, 2011). That the derivatives are available opens up the way for methods like the interval Newton contraction methods explained earlier this chapter.

4-7 Conclusion

In this chapter the basic principles that apply when using interval analysis are discussed. After explaining the basic arithmetic operators the concept of dependence was mentioned. To reduce the effect of dependence it is important to keep the number of instances of a parameter as small as possible in a function expression. Furthermore a general approach to interval analysis as an optimization tool has been explained. It is also mentioned how

zero finding interval algorithms can be used to find aircraft trim points. One of the main parts of interval optimization algorithms are box contraction methods. Newton's method, box consistency and hull consistency have been described. The most important issue of using interval analysis with B-spline models has been highlighted as well. With all of this covered the next chapter starts with applying interval analysis to solve optimization problems.

Chapter 5

Preliminary analysis

This chapter will show the optimization power of interval analysis by means of some practical examples. First it is shown that it is indeed capable of finding the minimums in functions that other methods have great difficulties with. The effect of box splitting strategies are also elaborated upon.

5-1 Optimization power of interval analysis

To show the power of interval analysis optimization benchmark problems are used. The Easom function is one of these benchmark problems. It is a two dimensional function that is defined on a square with the ranges $-100 < x_1 < 100$ and $-100 < x_2 < 100$. The function definition is given in Equation 5-1. The shape of the function is shown in Figure 5-1. The true minimum is located at (π, π) and its value is $y(\pi, \pi) = -1$. In Figures 5-2 and 5-3 one can see what an interval analysis solution to this problem would look like. It can be seen that the number of boxes keeps growing exponentially until the fifteenth iteration. It is impossible to see but at that iteration the lowest upper boundary of a box became marginally lower than the constant value that the algorithm had found until that moment. Apparently a lot of boxes had a lower bound that was higher than the new slightly lower estimate of the minimum. This caused the massive drop in the number of boxes. In the figure showing the box locations it can be seen that after the fifteenth iteration the remaining boxes cluster around the solution, becoming ever narrower as the iterations go on. The final solution that was found with this algorithm is a minimum of -1.0000 contained by the box $[x_1] = [3.1415, 3.1416], [x_2] = [3.1415, 3.1416]$. The box is very small, and also does contain the true location (π, π). This shows that interval analysis is capable of finding the minimum value of a function that has highly nonlinear parts.

$$y(x_1, x_2) = -\cos(x_1)\cos(x_2)e^{-(x_1-\pi)^2-(x_2-\pi)^2} \tag{5-1}$$

In (Van Kampen, 2010) the results from 100 runs of a constrained and an unconstrained gradient based optimization are given. Furthermore the results for 100 constrained genetic

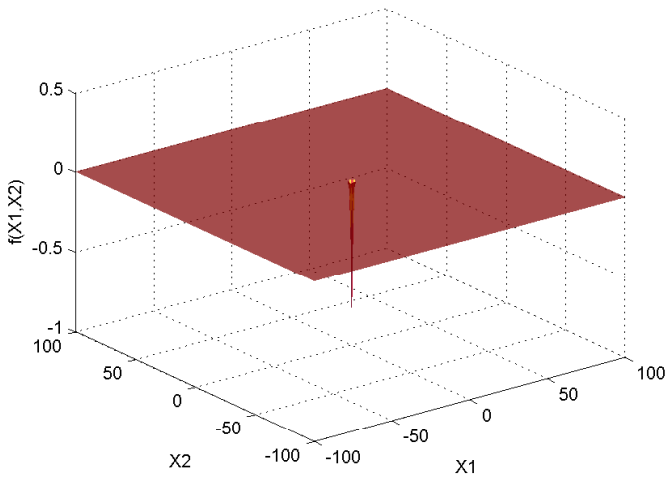

Figure 5-1: The Easom function

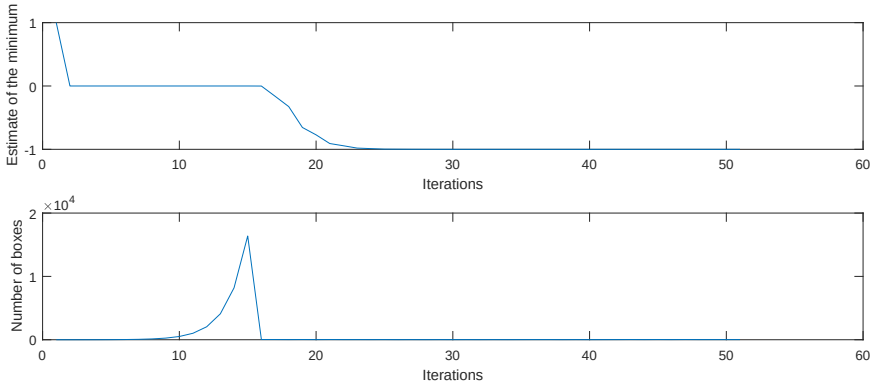

Figure 5-2: Estimation of the minimum and the amount of active boxes during the solve for the Easom function

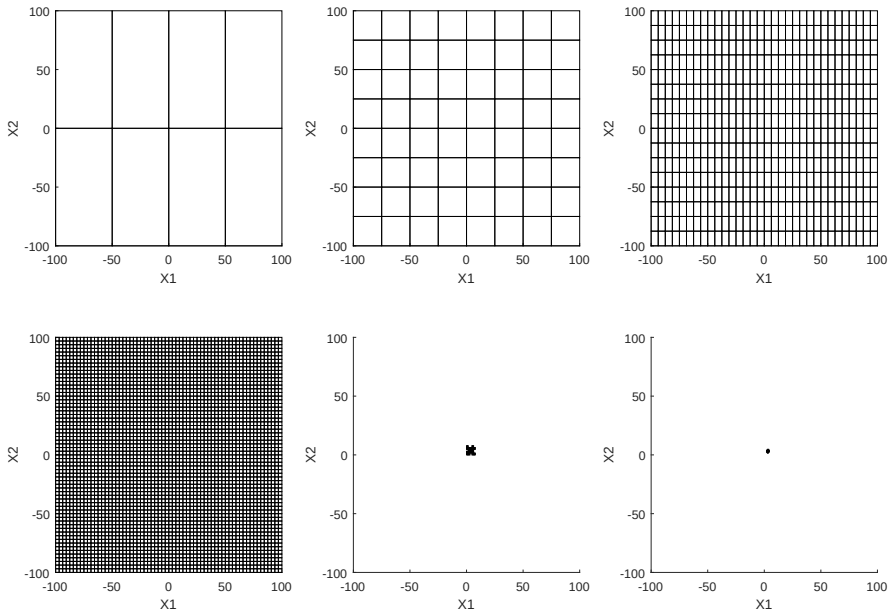

Figure 5-3: Development of the boxes during the solve for the Easom function

algorithm runs are given as well. These are shown in Figures 5-4 and 5-5. It can be seen that none of the gradient based results actually find the minimum. This is because the Easom function is almost completely flat for most of the domain. The genetic algorithm performs significantly better, but still only half of the runs find the minimum. This again shows the power of interval analysis, since it only needs one run to find the true minimum, where other methods might not even find it at all when allowed multiple runs.

5-2 Box splitting strategy

In the field of interval analysis there is not really a consent of what direction to take when splitting the boxes during an optimization problem. Here three methods are compared: random direction, the dimension that is the largest and the direction that gives the greatest improvement. The method of random splitting is very straight forward. Just pick a random box dimension and split that. The method where the largest dimension is used for splitting requires just a simple check to find out what dimension of the box is largest. This dimension is then used for splitting. This method will prevent boxes becoming very slender by splitting in the same direction over and over again. Finding the direction of largest improvement is done as follows. All but one of the intervals making up the box are set to their mean value. This leads to n one-dimensional boxes when solving an n-dimensional problem. All these one-dimensional boxes are put into the cost function. The direction that gets the largest interval from the cost function is likely to provide the greatest improvement rate.

These three methods are tested on the three-dimensional Hartmann function. This function as given in Equation 5-2 is defined for $x_i \in (0, 1)$ for $i = 1, 2, 3$ and has four local minimums.

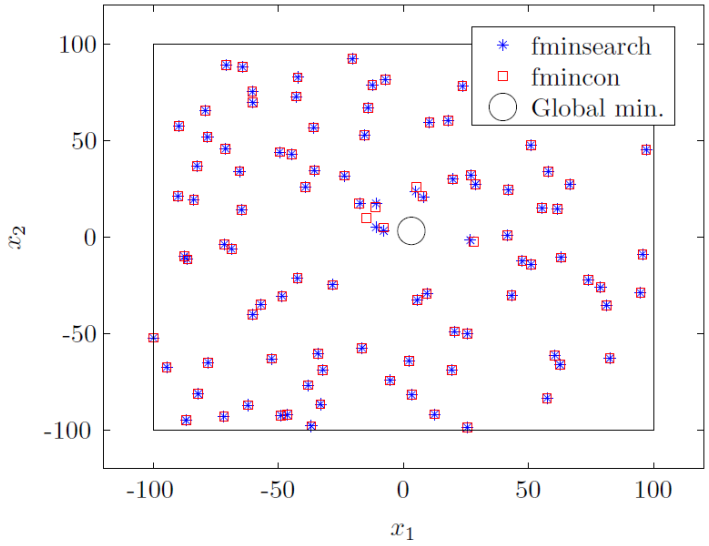

Figure 5-4: Solutions of constrained (`fmincon`) and unconstrained (`fminsearch`) gradient based methods for the Easom function (Van Kampen, 2010)

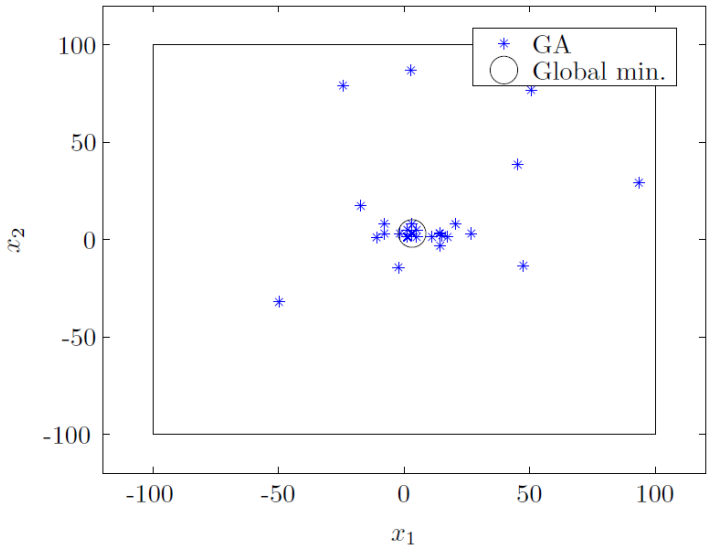

Figure 5-5: Solutions of a genetic algorithm for the Easom function (Van Kampen, 2010)

Table 5-1: Results from the Hartman problem when using different box splitting strategies

	Hartmann	Random	Largest dimension	Greatest improvement
Minimum	-3.86278	-3.8586	-3.8483	-3.8595
x_1	0.114614	$[0, 0.71875]$	$[0, 0.33594]$	$[0.015625, 0.21875]$
x_2	0.555649	$[0, 1]$	$[0.52637, 0.58398]$	$[0.54199, 0.56836]$
x_3	0.852547	$[0.625, 0.9375]$	$[0.83398, 0.86914]$	$[0.84424, 0.86035]$

Figure 5-6 shows how the solution developed with each iteration. The results of the minimization problem are shown in Table 5-1. When obtaining these results no stopping criteria were set in terms of accuracy. The calculations stopped after 30 iterations, in order to make the comparison as fair as possible. When looking to the figure one can easily see that the approach where the greatest improvement is used as a splitting direction converges to the minimum value the fastest. More important than the faster convergence is that greatest improvement splitting requires much less boxes for the same number of iterations. The containment box for the random splitting is very large compared to the boxes found by the other splitting methods. This happens because the final box is determined by taking the hull of the remaining boxes. If the random splitting code splits a box that contains some low cost function values in the same direction over and over again, this box becomes very slender, but will not be deleted since it does contain some low cost values. If the hull of the remaining boxes is taken such a slender box creates a large area that the code already knows not to contain the solution. To illustrate this a similar experiment is performed on the two-dimensional Levy function, as defined in Equation 5-3. The results after 20 iterations are shown in Table 5-2 and Figures 5-7 to 5-10. The performance of the different splitting strategies shows the same trends for the Levy function as for the Hartmann problem. The hull of the solution boxes is the red box in the figures. For the random splitting method a lot of previously discarded space is included.

$$
\begin{aligned}
f(\mathbf{x}) &= -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right), \text{where} \\
\alpha &= (1.0, 1.2, 3.0, 3.2)^T \\
(A) &= \begin{pmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix} \\
(P) &= 10^{-4} \begin{pmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{pmatrix}
\end{aligned}
\tag{5-2}
$$

$$
\begin{aligned}
y(x_1, x_2) &= \frac{\pi}{2}\left(10\sin\left(\pi\left(1 + y_1\right)\right)^2 + y_1^2\left[1 + 10\sin\left(\pi\left(1 + y_2\right)\right)^2\right] + y_2^2\right), \text{where} \\
y_1 &= \frac{x_1 - 1}{4} \text{ and } y_2 = \frac{x_2 - 1}{4}
\end{aligned}
\tag{5-3}
$$

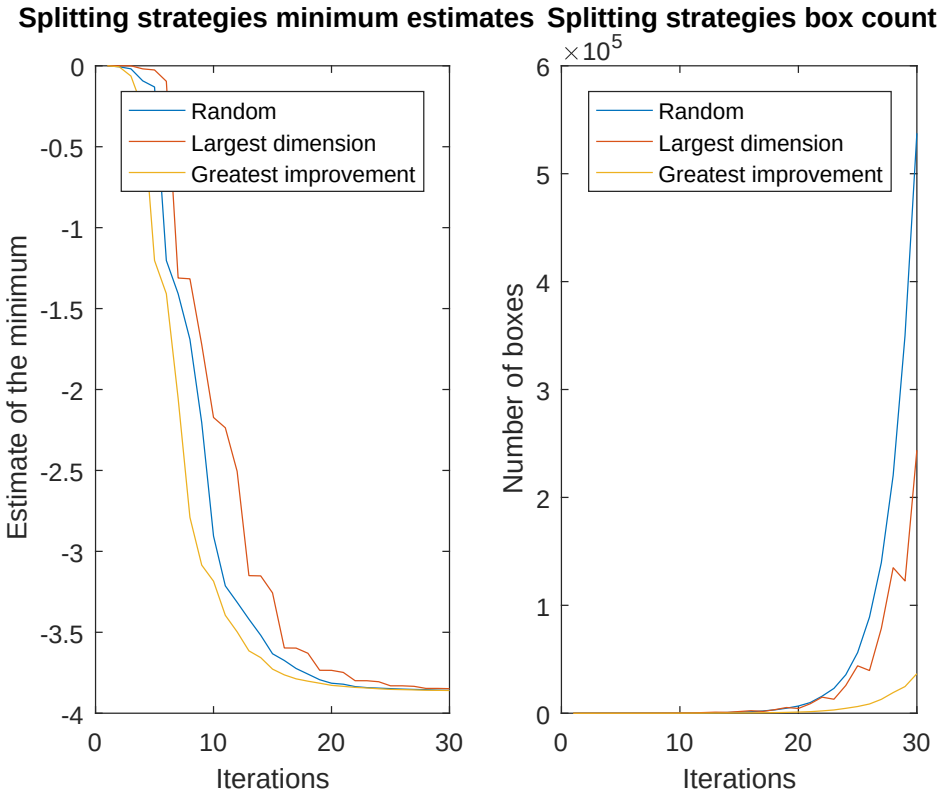

Figure 5-6: Development of the solution for the Hartmann problem when using different box splitting strategies

Table 5-2: Results from the Levy problem when using different box splitting strategies

	Levy	Random	Largest dimension	Greatest improvement
Minimum	0	0.0023	0.0024	$7.1961e^{-4}$
x_1	1	$[0, 1.2501]$	$[0.9765, 1.0352]$	$[0.9863, 1.0157]$
x_2	1	$[0.8398, 1.1524]$	$[0.8203, 1.1719]$	$[0.8593, 1.0938]$

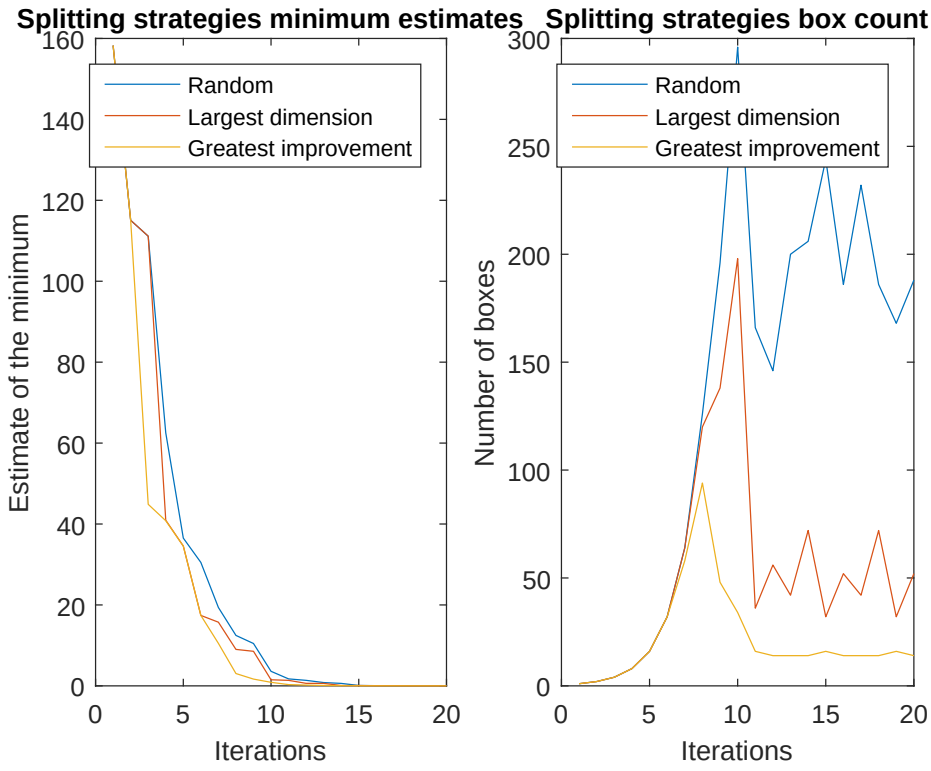

Figure 5-7: Development of the solution for the Levy problem when using different box splitting strategies

Figure 5-8: Solution boxes of the Levy function with random box splitting

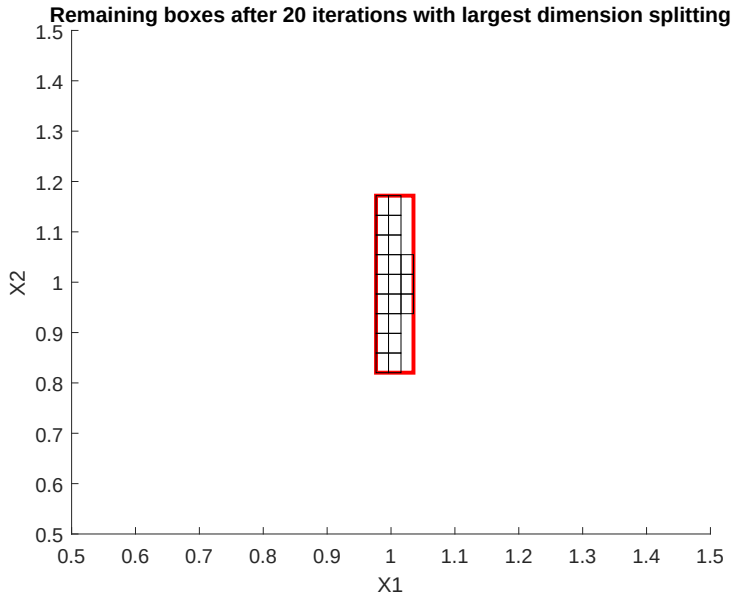

Figure 5-9: Solution boxes of the Levy function with largest dimension box splitting

Figure 5-10: Solution boxes of the Levy function with largest size box splitting

5-3 Conclusion

This chapter has shown that interval analysis is indeed capable of finding the minimum of a nonlinear cost function, even where other optimization methods fail. Three different box splitting methods have been tested and compared. Using the direction that yields the greatest improvement gave the best results and is therefore the favorable approach to be implemented in interval analysis optimization routines in the rest of this work. Comparing the results of the different splitting methods revealed a big challenge ahead when documenting the found trim set of the ICE model. The boxes cannot simply be joined by taking the hull of all boxes because a lot of states that are not actual trim states will be included. This would lead to an incorrect trim set that should not be used for further research on the ICE model.

Chapter 6

Converting ICE

As mentioned earlier in Chapter 4 the interval toolbox Intlab is not compatible with Simulink. This means that the model must be converted into an equivalent form that is compatible with Intlab. This proces is discussed in this chapter. Converting a Simulink model to a Matlab code equivalent is a big task. But it also gives the freedom to implement it in such a way that it is very effective for the task at hand. In the Simulink model there are a lot of feedback loops that provide the model with parameters that are changing with flight conditions. Implementing something like that in code would result in a model that needs to iterate until it finds a stable situation. Since interval analysis is computationally heavy as it is this is highly undesirable. Instead these feedback variables are computed beforehand and fed into the model. This gives a lot of freedom in setting the flight conditions, since a lot of flight parameters can simply be set to the desired value. A disadvantage of this approach is that the final code version of the model can only be used for trimming and not for flight simulation.

Previously, trim has been defined as the states where all accelerations on the aircraft are equal to zero. So the overall goal of the trim model should be to compute accelerations. Since the mass and inertia of the aircraft are constant in this model, they can be directly calculated from the forces and moments acting on the aircraft body by applying Newton's second law. What the forces and moments are will be determined by controlling some of the states and inputs from the state and input vectors given in Equations 6-1 and 6-2. Using all variables in these two vectors as independent variables leads to an unnecessary large computational problem. To prevent this some of these variables can be fixed, so that they become constraints. As explained in (De Marco et al., 2007), choosing the flight condition to trim at influences what variables needs to be solved for. The rotational rates p, q, and r can be only constant values in trim, which makes it convenient to use them exclusively as constraints. By doing so one can find specific flight scenarios, which can be useful for a multitude of reasons. By choosing the flight conditions the Euler angles ϕ, θ, and ψ can be linked to other flight variables. This makes them into dependent variables. In the model the positions N and E are not really needed, because a homogeneous spherical gravity model of earth is assumed. Position on earth of the aircraft does not have any influence at all because of that. Finally there is the

thrust T. As aircraft trim is the target it is convenient to couple the required thrust to the body forces it needs to balance. How this is done will be explained later on. When using interval analysis the independent variables must be given as intervals, while the constraints are crisp numbers.

$$\mathbf{x} = [\alpha\ \beta\ M\ p\ q\ r\ N\ E\ D\ \phi\ \theta\ \psi]^T \tag{6-1}$$

$$\mathbf{u} = [\delta_{lele}\ \delta_{rele}\ \delta_{llefi}\ \delta_{llefo}\ \delta_{rlefi}\ \delta_{rlefo}\ \delta_{lamt}\ \delta_{ramt}\ \delta_{lssd}\ \delta_{rssd}\ \delta_{pf}\ \delta_{PTV}\ \delta_{YTV}\ T]^T \tag{6-2}$$

6-1 Required functional blocks

Referring to the Simulink model in Figure 2-4 it can be seen that there are five functional blocks: Aerodynamic Coefficients, Forces and Moments, 6DoF (Euler Angles), Environmental Models and Alpha, Beta, Mach. It will now be explained how these blocks are implemented in the Matlab code version of the ICE model.

6-1-1 Alpha, Beta, Mach

A lot of the functionality of the Alpha, Beta, Mach block is required in order to make the feedback loop in Simulink work. Because the rates, Mach number and incidence angles are set either as independent variables or constraints they do not need to be calculated. Furthermore in the inputs to the block there are wind velocity and wind angular rates. Trim points are steady states, so no wind functionality is required. As such wind can be removed from the model completely. The only two things that need to come from this block are the dynamic pressure $\bar{q}$ and the airspeed V. Mach number and altitude are known from the input to the trim procedure. The altitude can be used to calculate the atmospheric density, that in turn can be used to calculate the speed of sound at that altitude. With the Mach number known the airspeed is easily computed. This can then be combined with the earlier calculated density to determine the dynamic pressure. There is no particular reason against splitting up the functionality of blocks, and here it has been used to remove the need to calculate two variables in the Environmental Models block. The biggest advantage of this is that the dynamic pressure $\bar{q}$ and the airspeed V can now be calculated without the need for a feedback loop.

6-1-2 Aerodynamic Coefficients

The Aerodynamic Coefficients block contains the aerodynamic model of the aircraft. The has thrust as an input to the block, but when looking at the implementation of the block it is not used anywhere. As such it is left out of the Matlab code version, which also makes it easier to couple the required thrust with the body forces acting on the aircraft. Only the eleven control surface deflections are used in this block. The set op inputs is then used to interpolate the data tables. Since the independent variables are intervals interval interpolation

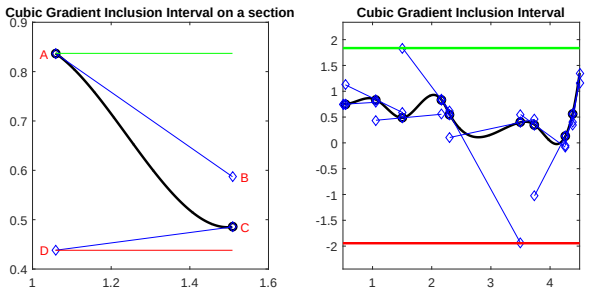

Figure 6-1: Illustration of the cubic gradient inclusion interval

must be used. The approach to linear interpolation has been explained in Chapter 3. This is enough for 96 out of 108 data tables. The other twelve require cubic interpolation. To get an inclusion interval for the cubic interpolation curve the gradients at the data points are used. That is why it is called the cubic gradient inclusion interval. It is explained below. In the current implementation a nummeric approximation for the gradient is used. If an analytical expression for the gradient is available that will work just as well. From the interpolations come intervals for the aerodynamic force and moment coefficients that are needed for the Forces and Moments block.

Cubic Gradient Inclusion Interval

The second derivative of a cubic spline between two data point is at most a linear function. Therefore there can be at most one zero crossing for the second derivative, hence there can be at most one inflection point between two data points. If the tangent is taken at the data points and extended to the other side of the section an inclusion interval can be determined. When there are no crossings of the tangent and the curve the endpoints of the tangent are measures for the inclusion interval of the curve. If the curve intersects the tangent line before it reaches the other side of the section this means that there has to be an inflection point between the data point at which the tangent was taken and the intersection of the tangent and the curve. This means that the curve can only continue in the direction that it has crossed the tangent. If the tangent is taken from the left data point it runs from point the data point A to the mapped point B. The tangent from the right data point C runs to the mapped point D. It is certain that the curve is included in the interval $[min(A, B, C, D), max(A, B, C, D)]$. The left plot in Figure 6-1 shows this procedure. When this is done over all the sections defined by data points and domain boundaries an inclusion interval for the entire domain can be found by taking the maximum and minimum found values, as is illustrated in the right plot in Figure 6-1. The curves used to make the figures are based on a cubic spline created on a set of ten random numbers in the range $[0, 1]$ randomly distributed on the domain $[0.5, 4.5]$.

6-1-3 6DoF (Euler Angles) and Environmental Models

The 6DoF block is used in the model to integrate forces and moments to velocities and positions. It also applies Newton's second law to convert them to accelerations. The second moment integration results in the Euler angles ϕ, θ, and ψ.

If the interval results are used here, this means that intervals have to be integrated twice to calculate them this way. Integrating intervals generally leads to very wide intervals (Neumaier, 2003), (Kraemer, 2006). This is clearly undesirable so instead the flight condition is used to determine the Euler angles. Before explaining how that is done it is necessary to determine what they are needed for. There is only one part of the model that depends on the Euler angles and that is the conversion of the gravity force from the earth axis system to the body axis system. To indicate how convenient this actually is the calculations that need to be done should be shown. Equation 6-3 shows the rotation matrix from the earth reference frame to the body reference frame, while Equation 6-4 show the gravity vector in the earth reference frame. When calculating the gravity force in the body reference system the zeros in the gravity vector make the first two columns of the rotation matrix obsolete. This simplified calculation of the gravity force is shown in Equation 6-5. It is important to note that the yaw angle ψ is now completely removed from the equation. Its value can therefore be chosen arbitrarily and will not affect the calculations in any way.

$$R = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\cos\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \tag{6-3}$$

$$F_{g,e} = \begin{bmatrix} 0 \\ 0 \\ g \cdot m \end{bmatrix} \tag{6-4}$$

$$F_{g,b} = \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} \cdot g \cdot m \tag{6-5}$$

So what is left to be done is to determine a pitch and a roll angle. The choice of flight condition influences how this should be done. A useful trim state would be straight and level flight. This would imply that the pitch angle θ is equal to the angle of attack α and that the roll angle ϕ is equal to zero (De Marco et al., 2007). Since the angle of attack is one of the inputs to the trim routine this is a feasible option. However, the ICE model has a little peculiarity that its aerodynamic model is not symmetrical. This results in a small side force when trimming to a zero-roll angle state. The flexibility gained by implementing the model just for the purpose of trim gives rise to a solution. It is possible to calculate the aerodynamic forces and moments on the aircraft before determining the gravity force contribution. Therefore the side force can be used as an input to the gravity calculation function. In this function Equation 6-5 is implemented. By enforcing that the sum of the side force and the contribution of the gravity force should be equal to zero in trimmed flight the equation for the force in y-direction can be inverted to calculate the roll angle. This is shown in Equation 6-6. When trimming for turn scenarios the desired turn rate will lead to a required side force to sustain the turn. This side force can be used to determine the roll angle in the exact same way. The most important implication of this is that no outputs of the 6DoF block are needed for any of the other blocks, removing all the feedback loops.

$$\phi = \sin^{-1}\left(\frac{-F_{y,aero}}{\cos(\theta) \cdot g \cdot m}\right) \tag{6-6}$$

This shows how the gravity force can be determined. Earlier in the explanation of the Alpha, Beta, Mach block it was argued that the wind models can be removed since this converted version of the model is only going to be used for trim. That and the choice to put the calculation of the speed of sound and the air density in the functionality of the Alpha, Beta, Mach block results in the Environmental Model not needing any inputs from the 6DoF block. This means all feedback loops are now removed.

6-1-4 Forces and Moments

The Forces and Moments block takes the aerodynamic coefficients and turns them into body forces and moments. The thrust vectoring deflections are used to calculate additional moments. The aerodynamic side force is used to determine the required roll angle. With that angle known the gravity force can be calculated as was explained above. The thrust of the model is only acting along the body x-axis, and as such the sum of the x-components from the aerodynamic and the gravity forces can be used to find the required thrust. This works because the model will only be used for trim. The found thrust is used to calculate the moments caused by the thrust vectoring. Finally accelerations are determined by applying Newton's second law with the aircraft mass and inertia measures.

6-1-5 Overview of the converted model

To clarify the explanation above the structure of the converted model is given in Figure 6-2. It is clear to see that there are no feedback loops required to calculate the accelerations. The output of this model is used in the interval analysis trim algorithm.

6-2 Model validation

To ensure that the converted model is the same as the original model it must be validated. This can be done by generating a large set of random inputs and comparing the results of the two models. If the differences are small the model can be accepted and used for trimming purposes. Comparison is done block by block referring to Figure 6-2. Based on the input the atmospheric properties block calculates the airspeed V and dynamic pressure $\bar{q}$. Internal variables for speed of sound a and air density ρ are stored as well. Because the thrust is required for the calculation of the aerodynamic coefficients in the Simulink version of the model the thrust finding functionality of the converted model is disabled. Instead it is changed slightly to also be capable of taking thrust as an input. It is set as one of the randomly generated input variables for the validation process. The coefficients are calculated for both models and the difference of the two is stored. These are intervals, because the converted model is build to output intervals. The midpoint of these intervals should be close to the crisp output of the Simulink model, while the width should be small. The midpoints of the interval coefficients are used to evaluate the forces and moments blocks in both models. This is to prevent continuation errors should there be a significant difference. The difference between the forces and moments are saved. Based on the forces the Euler angles can be calculated. Because the inputs are completely random it is possible that the roll angle cannot be large enough. In that case it becomes an imaginary number. This is unusable for the rest

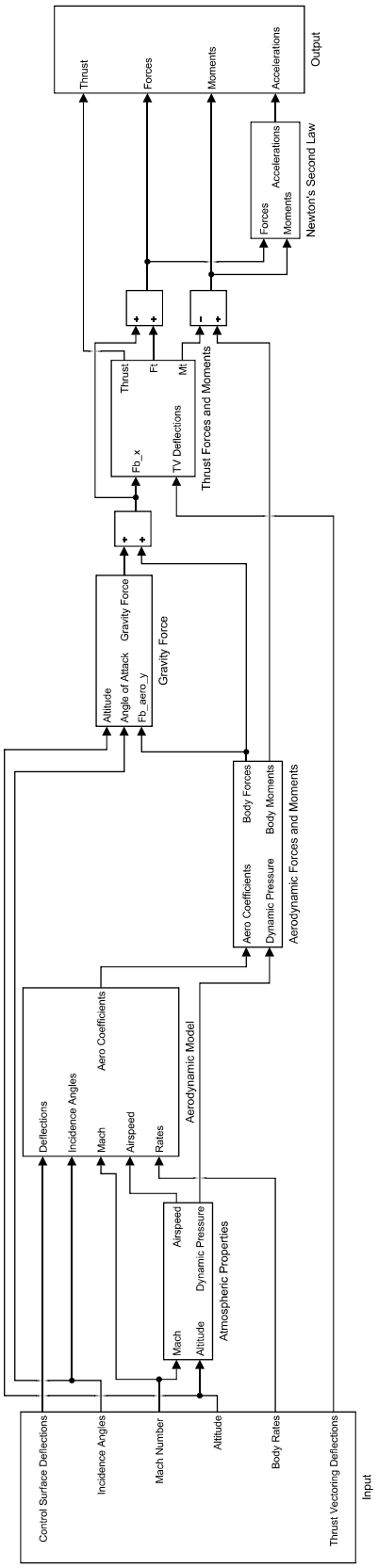

Figure 6-2: Implementation of the ICE model in Matlab code so it can work with Intlab

of the process, so if a combination of inputs is found to generate nonreal Euler angles that combination is discarded. If the number of combinations is large enough this does not create a problem for the overall validation process. If the Euler angles are real the gravity force block also gives the direction cosine matrix, the Euler angles, and the gravitational acceleration next to the gravity force. The Euler angles are used as an input to the equivalent system in Simulink, again to prevent continuation errors. This system also calculates atmospheric properties calculated in the beginning of each step. These can now also be compared. Note that in this validation process only forces and moments are used for comparison. Accelerations are simply linearly dependent on the forces and moments, so including them would not give any additional information.

To create the validation data 10000 randomly generated input combinations were made. Of those 10000 there were 1446 combinations that yielded nonreal roll angles, which means that there are 8554 usable samples. This is more than enough to do a statistical analysis of the results. An overview is given in Table 6-1. It is clear to see that all differences are very small, as well as the variations in the differences. The largest differences still are several orders of magnitude smaller than the actual values, so it can be concluded that the converted model is a good representation for the ICE model and that it can be used to find the trim set.

6-3 Conclusion

In this chapter an extensive description has been given of the steps taken to convert the Simulink ICE model to a Matlab equivalent. Because this was done with the purpose of trim in mind some of the original functionality is lost. Instead there is a lot more control over the flight state in the new version. Comparing the two models has shown that there are only very small differences between the two models, so the converted model can safely be used to find the trim set of the ICE model.

Table 6-1: Statistical results of the differences between the models

Variable	Mean	Standard deviation
Midpoint F_x coefficient	$9.6541 \cdot 10^{-19}$	$1.3406 \cdot 10^{-17}$
Midpoint F_y coefficient	$-6.0855 \cdot 10^{-20}$	$8.5915 \cdot 10^{-18}$
Midpoint F_z coefficient	$-5.7949 \cdot 10^{-18}$	$2.5015 \cdot 10^{-16}$
Midpoint M_x coefficient	$-9.7342 \cdot 10^{-20}$	$7.2388 \cdot 10^{-18}$
Midpoint M_y coefficient	$-1.8790 \cdot 10^{-20}$	$1.8622 \cdot 10^{-17}$
Midpoint M_z coefficient	$-2.3416 \cdot 10^{-19}$	$2.0915 \cdot 10^{-17}$
Diameter F_x coefficient	$4.9780 \cdot 10^{-17}$	$3.3061 \cdot 10^{-17}$
Diameter F_y coefficient	$3.0497 \cdot 10^{-17}$	$2.3705 \cdot 10^{-17}$
Diameter F_z coefficient	$1.5096 \cdot 10^{-15}$	$1.0107 \cdot 10^{-15}$
Diameter M_x coefficient	$1.8256 \cdot 10^{-17}$	$1.6159 \cdot 10^{-17}$
Diameter M_y coefficient	$9.3947 \cdot 10^{-17}$	$7.3954 \cdot 10^{-17}$
Diameter M_z coefficient	$7.0274 \cdot 10^{-18}$	$5.4121 \cdot 10^{-18}$
F_x	0	0
F_y	0	0
F_z	0	0
M_x	0	0
M_y	0	0
M_z	0	0
F_{g_x}	$7.1055 \cdot 10^{-13}$	$1.5572 \cdot 10^{-12}$
F_{g_y}	$-1.1246 \cdot 10^{-14}$	$1.3368 \cdot 10^{-12}$
F_{g_z}	$-1.7949 \cdot 10^{-12}$	$3.2110 \cdot 10^{-12}$
g	$-1.8503 \cdot 10^{-15}$	$2.5640 \cdot 10^{-15}$
a	$-1.7473 \cdot 10^{-13}$	$1.0449 \cdot 10^{-13}$
ρ	$3.8953 \cdot 10^{-12}$	$1.9601 \cdot 10^{-12}$
M	$2.0968 \cdot 10^{-16}$	$1.8797 \cdot 10^{-16}$

Chapter 7

Additional results

Next to the results that have been explained in the paper more results have been produced, using more or different control effectors. This chapter will go over those results. First a larger set of point solutions is discussed. In this set there are results for a third altitude of 30,000 ft, as well as results that are not on the Mach numbers used in the paper. There is also a set using the left SSD instead of the AMT, again at the three different altitudes. The point solutions indicate that a demonstration of the power of interval analysis can be done by locking the left AMT deflection to 7°, while solving for angle of attack, Mach number and elevon deflections. Next are results showing how trim sets can be extended by using more control effectors, so that redundant control power can be generated. Each scenario used a fixed angle of attack of 5° with Mach number and elevons free to assume any value in the domain. The first configuration extended this with both AMTs, the second with both SSDs, the third with the left AMT and the left SSD and the last configuration with the left AMT and the right SSD.

7-1 More point enclosures using the LAMT to balance

In the initial phase of evaluating whether or not the code was converging to proper solutions two different approaches were tried. In the first approach Mach numbers were set so that the code had to find the required control deflections for the elevons and the left AMT. In the second approach angles of attack were set, so a value for Mach number had to be found. As it turns out there are trim states for every value within the allowable Mach number range, while there are quite some angles of attack that do not yield solutions. There is a small portion of the model that shows some odd behavior requiring an angle of attack as low as $-15.69°$ when trimming at 10,000 ft. Because that is outside the limits of the converted model this section is the only place where there are no solutions when using a fixed Mach number. Both approaches have been tried at 10,000 ft and 20,000 ft. After that it was realized that solving for α yielded better organized results that approach was used for a set of solutions at 30,000 ft. All points found in this phase of the research are shown in Figure 7-1. The third altitude

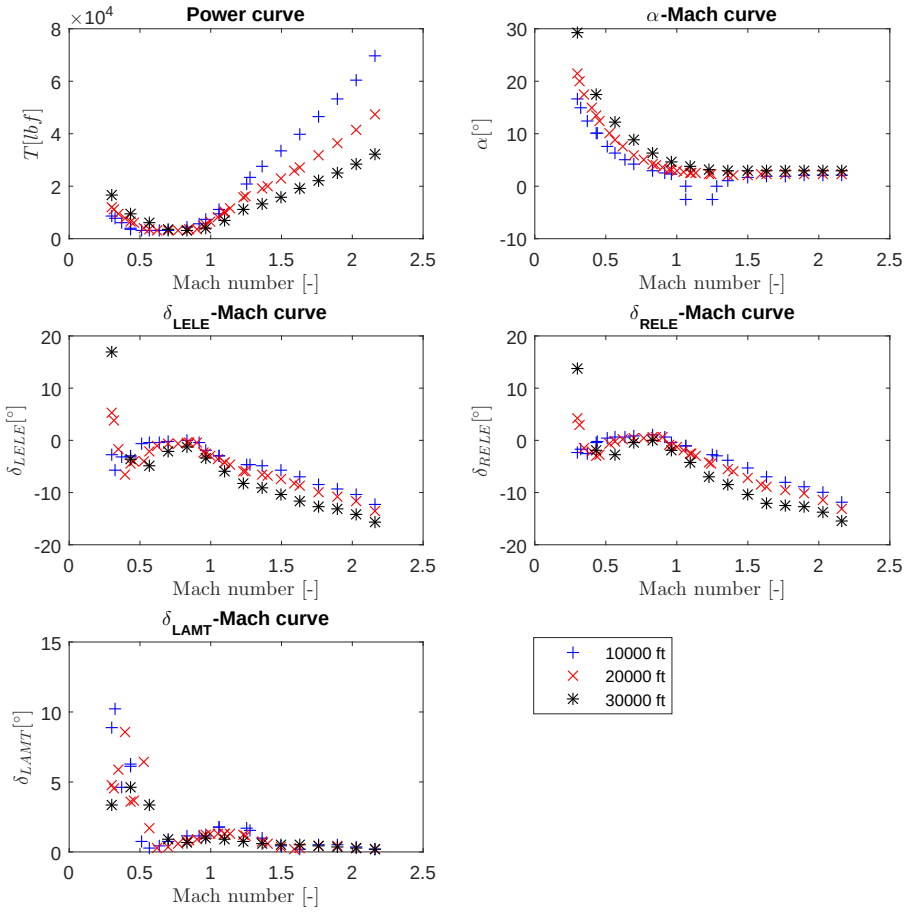

Figure 7-1: Overview of the found trim points when using fixed Mach numbers and the LAMT for balancing

adds information about the effect of altitude on the found points. The difference between 10,000 ft and 20,000 ft solutions already showed that increasing altitude moves the power curve to the right and rotates it clockwise, and the 30,000 ft points confirm this. The same goes for the α-M curve. At greater altitude larger angles are needed due to the thinner atmosphere and again the found results confirm this. Because of the thinner atmosphere it is also expected that the control deflections need to be larger to create similar moments, and again this is confirmed by the results. From the 10,000 ft and 20,000 ft it appeared that there is a coupling between the difference in deflection for the elevons and the left AMT deflection. This relation is less strong at 30,000 ft but at high speeds it certainly is there. From the runs that solved Mach number for a fixed angle of attack the odd behavior of the model at 10,000 ft could already be seen for $\alpha = 0°$ and $\alpha = -2.5°$. On the bright side did these runs confirm that the code finds multiple solutions when they exist.

It was shown earlier that only two points at 20,000 ft had accelerations that were smaller than the human detectable acceleration thresholds of $0.02 m/s^2$ for linear accelerations and

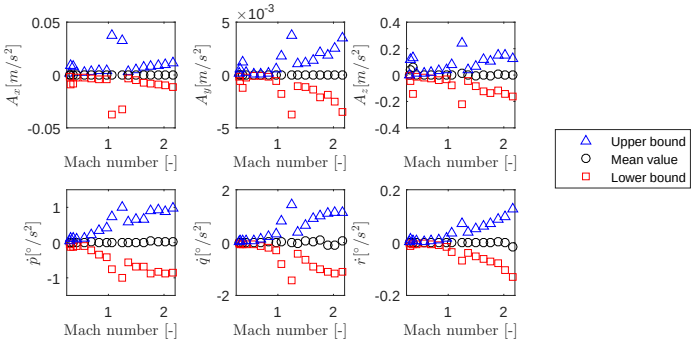

Figure 7-2: The hull of the leftover accelerations at 10,000 ft using the LAMT for balancing

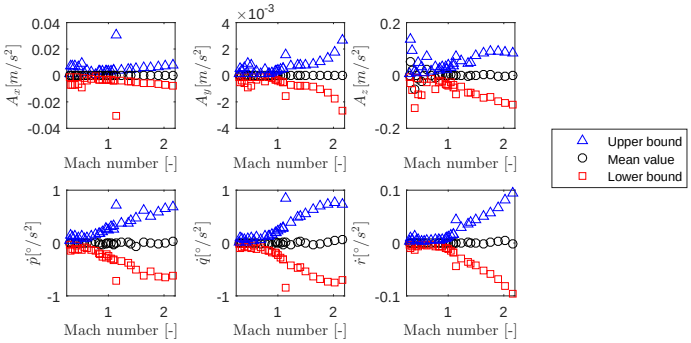

Figure 7-3: The hull of the leftover accelerations at 20,000 ft using the LAMT for balancing

$0.05°/s^2$ for rotational accelerations. Evaluating the accelerations of all the points from Figure 7-1 shows that there are only two additional points that can be fully accepted as trimmed. These are the points at the slowest two velocities at 30,000 ft. The hull of all accelerations can be seen in Figures 7-2 to 7-4. The trends known from earlier are present here as well. At low speeds the hull of the accelerations is much smaller than at high speeds. Increasing the altitude also reduces the hull width. This has to do with the the control deflection interval and the conversion to control forces and moments, as explained earlier. It is also interesting to know that solving for Mach number at fixed angle of attack resulted in much wider acceleration intervals. This gives another reason to keep working with the fixed Mach number approach.

7-2 Point enclosures using the LSSD to balance

In (Dorsett et al., 1996) and (Dorsett et al., 1997) the AMTs were ranked as the best all round yaw control effector. The second best are the SSDs. Because of this it is decided to find the three sets of trim points based on the fixed Mach numbers again, but instead of using the left AMT the left SSD is used. Again the final interval enclosures are small enough to take the average of the midpoints of the remaining boxes to create clearer plots. These are given in Figure 7-5. The power curves have the expected shape and also show the influence of altitude correctly, shifting to the right and leaning back as altitude increases. In the α-M

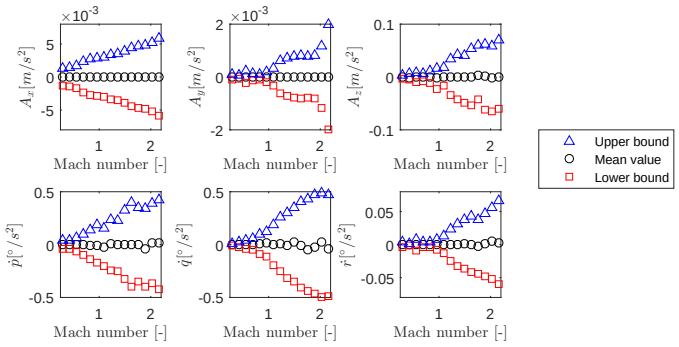

Figure 7-4: The hull of the leftover accelerations at 30,000 ft using the LAMT for balancing

curves the required angle of attack goes up for a given Mach number as the altitude increases. Once more this is as it should. These are important observations because they lead to the conclusion that the choice control effector does not influence the algorithms ability to find trim sets. It should also be noted that the odd behavior of the model in the low sonic region at 10,000 ft is also present in this control layout. This again causes two points to be missing form those sets.

The required deflection of the left SSD shows comparable behavior as the AMT used before. In the region of the domain corresponding to the backside of the power curve deflections are significantly larger than in the rest of the domain. What is different is that instead of having close to zero deflection at high Mach numbers like the AMT the SSD remains deployed more. This can be expected as the AMT is more exposed to the freestream air, as opposed to the SSD that is integrated in the middle of the wing. In order to still have the balancing effect the latter needs to have a larger deflection. The relation between the SSD deflection and the difference between the elevon deflections appears to be present again, although it does show more complex behavior. Below Mach 1.3 it is a similar relation as the AMT had, above that Mach number it changes sign. Because the SSD in right in front of one of the elevons it is clear that there are some interactions going on that cause this.

To evaluate the quality of the found results the accelerations must be checked again. There are a total of four points that have accelerations from all boxes lower than the detectable limit of the human vestibular organ. The truly trimmed points are again the two lowest velocities at 20,000 ft and 30,000 ft. The conversion from control effector deflection to control force is the same for each control effector, so the same explanation holds here as to why the hulls of the accelerations become larger as speed goes up and altitude goes down.

7-3 Demonstrating the power of interval analysis

In Figure 7-1 it can be seen that there is a solution at 20,000 ft that has a left AMT deflection that is greater than 7°, while the two solutions around it require deflections that are smaller than 7°. This implies that there should be at least two solutions that have a left AMT deflection that is exactly 7°. A run that solved for elevons, angle of attack and Mach number with the left AMT fixed at 7° showed that there are in fact four trim points with that condition. Because these findings did not match with the interpolations of the previously

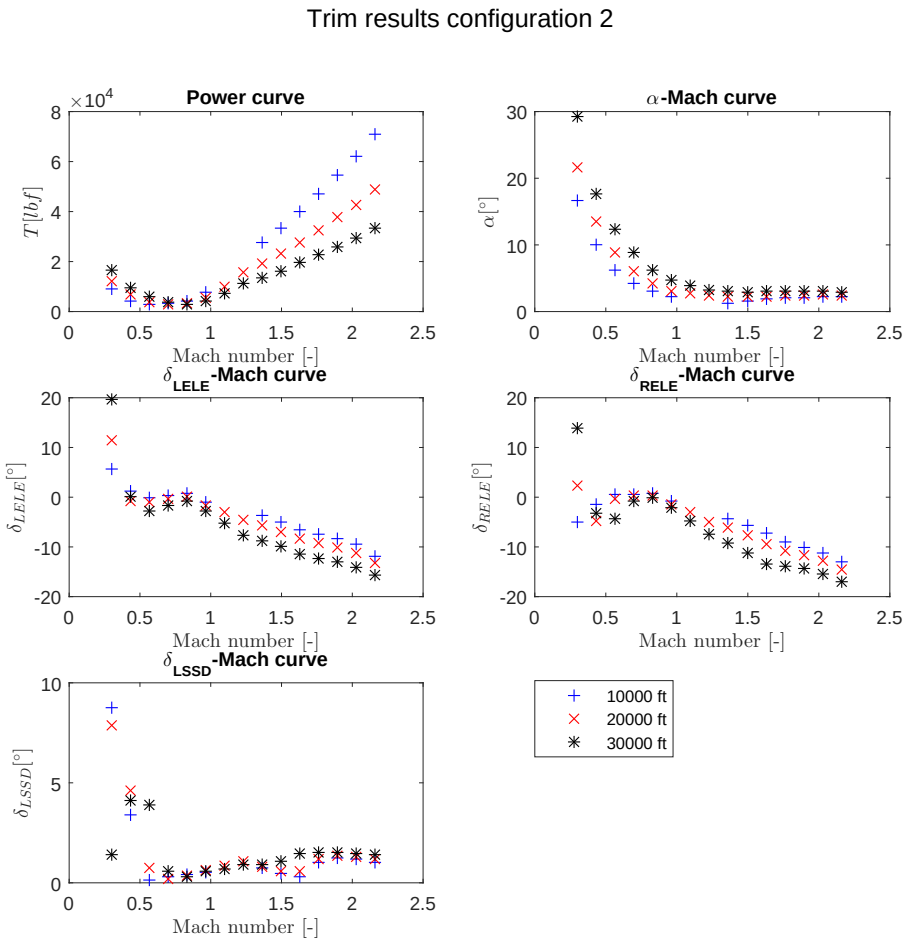

Figure 7-5: Overview of the found trim points when using fixed Mach numbers and the LSSD for balancing

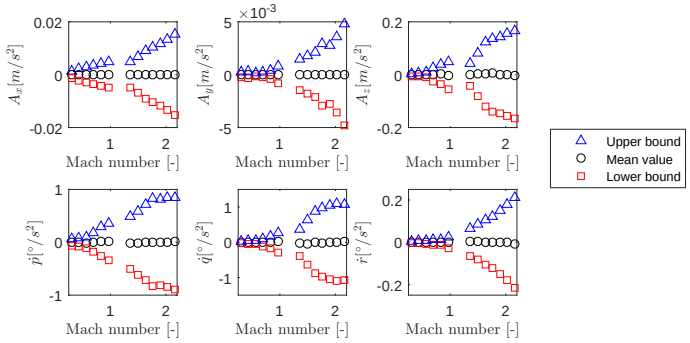

Figure 7-6: The hull of the leftover accelerations at 10,000 ft using the LSSD for balancing

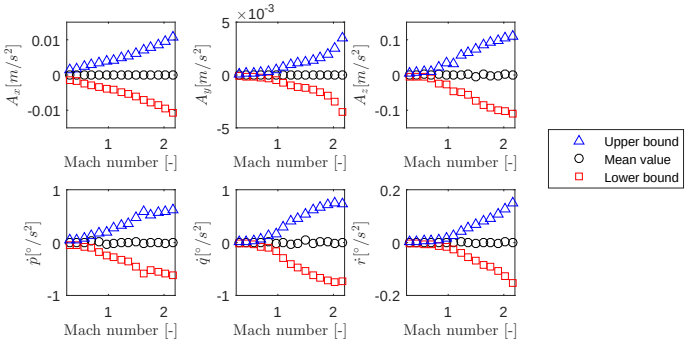

Figure 7-7: The hull of the leftover accelerations at 20,000 ft using the LSSD for balancing

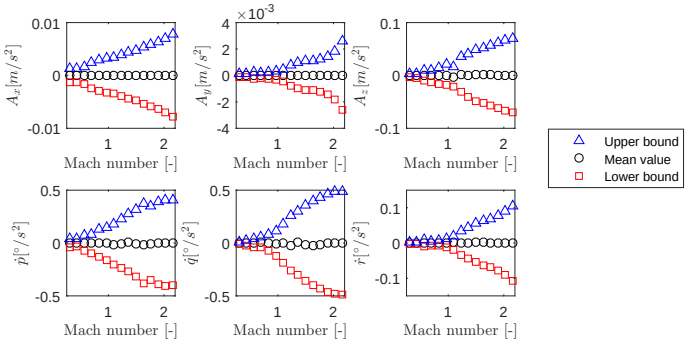

Figure 7-8: The hull of the leftover accelerations at 30,000 ft using the LSSD for balancing

found points a locally increased resolution of Mach number points was used. In the Mach domain between 0.35 and 0.55 points are spaced 0.05 apart. This gives 21 point enclosures that solve for α, elevon deflection and left AMT deflection. The results with the increased resolution show that there are indeed 4 solutions with a left AMT deflection of 7°. This is shown in Figure 7-9. This example has demonstrated that interval analysis is capable of finding multiple solutions in one run. The next step it checking if the interval analysis routine can also find continuous bounds on the same domain. Instead of providing 21 Mach numbers and doing 21 runs to find point enclosures there is now a Mach number interval $M = [0.35, 0.55]$. With this input a five variable optimization problem is created, as opposed to the four variable problems dealt with in all the above analyses. Figure 7-9 shows the bounds that have been found with this run, indicating that interval analysis is capable of bounding continuous solutions as well. As this run was intended as a demonstrator run for the power of interval analysis no review is done about the remaining accelerations.

7-4 Trim with redundant control effectors

So far trim sets have been found using the minimum required number of control effectors. In this section an evaluation will be made on how using more than the minimum amount of control effectors will effect the found trim sets. The large five variable runs shown earlier have demonstrated that a five variable problem is currently the limit that the code can handle

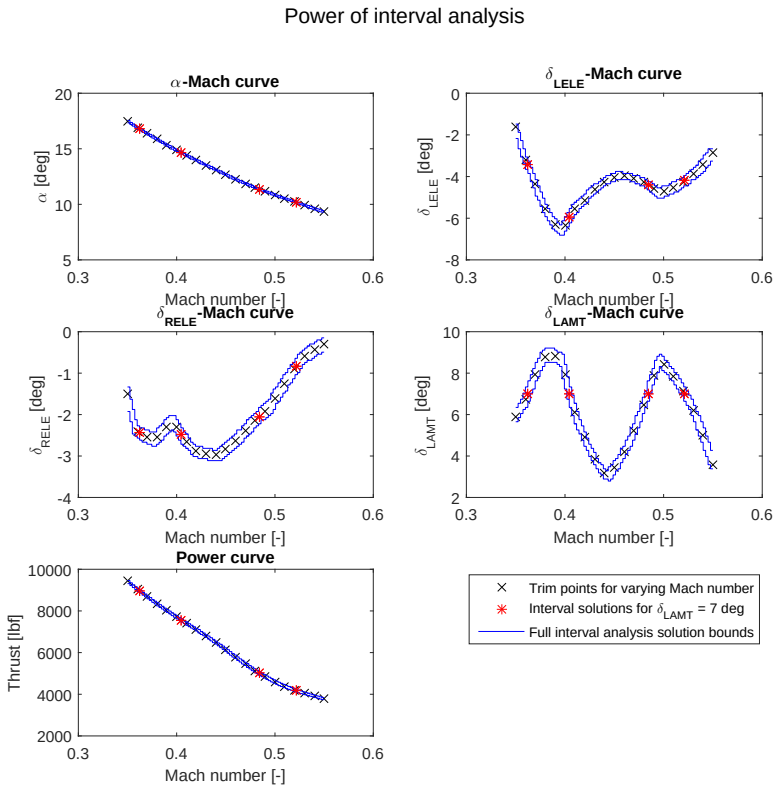

Figure 7-9: Results that show that interval analysis is capable to find multiple solutions at once

without taking an enormous amount of time. Using one additional control effector would lead to a six variable problem, so to prevent this the angle of attack is fixed and set to 5° for all runs in this section. The elevon deflections and Mach number are used as free variables in every run. In the first run this is extended with both AMTs and in the second with both SSDs. The last two runs use a combination of the left AMT with either of the SSDs. All runs are done at 20,000 ft, so the odd behavior of the model should not come into play here.

7-4-1 Both AMTs

The bounds found when using both AMTs are shown in Figure 7-10. It can be seen that there are two branches of solutions when plotted against Mach number. There are however unique solutions if plotted against thrust. This is done in Figure 7-11. There are two distinctive kinks in each plot. Both of these can be traced back to the AMTs data tables. When the AMTs exceed 10° the effect on the pitch moment increases less, while above 30° the moment effectiveness starts to decrease altogether. The elevons are deployed in the opposite direction to ensure that the moments remain neutral. The combination of AMTs and elevons now acts as a speedbrake. This increases drag for the first part of the curve, so the speed goes down. After the first kink the speed remains constaint, indicating that it is the slowest possible airspeed for the given control layout at $\alpha = 5°$. The drag does keep increasing though, as the required thrust keeps increasing in this section of the domain. After the second kink the airspeed increased again, because the elevons must be retracted to counter the reversed moment effectiveness of the AMTs. This decreases the speedbrake effect which reduces the drag increase, so the airspeed increases again. The bounds for the elevons are at most 0.35° apart, while for the AMTs 1.2° is the biggest difference between the upper and lower bounds. In the previous results the only trim points that met the acceleration limits occurred at low speeds, lower than the speeds found here. It is no surprise to find that no box has met all criteria for the accelerations. Figure 7-12 shows the leftover accelerations as a function of thrust. In this case there is not such a straight forward relation between airspeed and the width of these intervals. There are also some artifacts from sorting the data, most notably in the plot for A_z. Because the thrust value is calculated by the code and not an input the width of the thrust intervals is more stochastic than the width of the intervals defining the boxes. When generating these plots the list of boxes that must be evaluated for that instance of thrust varies continuously, causing the bounds of hull of the accelerations to oscillate as the box with the largest accelerations is removed and later replaced by a box that gives similar accelerations. The lower two plots in Figure 7-10 show the number of boxes and the relative box volume. The peak value of boxes was 43858 and the computation time was approximately 3 days, 5 hours and 23 minutes. Despite that this time is similar to the times used for the continuous trim sets discussed earlier the relative volume is only down to $2.4709 \cdot 10^{-8}$, while the other runs got below 10^{-10}. Because the different free variables the required number of box splits are different. This stops the calculations as early as iteration 25, while the other ones got as far as iteration 32. When evaluated at the same iteration the difference is only minor. This once more indicates that the choice of control effectors does not influence the effectiveness of the algorithm. Finally, it is to be expected that similar curves can be found at different angles of attack. Combined with the trim curves discussed in the paper, this will produce a large and complex trim set. There is no reason to assume that the method would not be capable of finding this set, except that the calculation time will be extremely long with

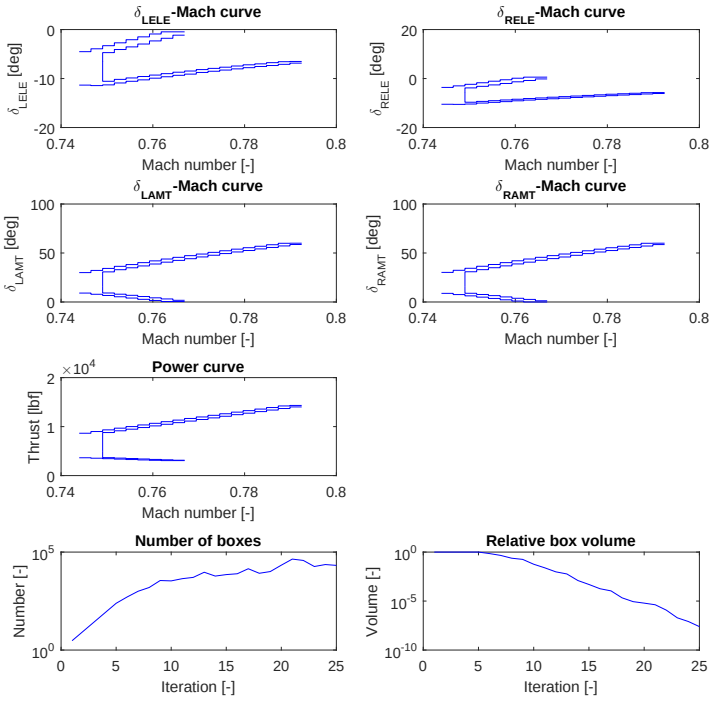

Figure 7-10: Trim enclosures as a function of Mach number when using both AMTs at $\alpha = 5°$

the current implementation. Because the solutions are expected to be a curve superimposed on another curve the solution will be a surface. Because of this it is expected that the required number of boxes will cause usable memory issues even on the best computers.

7-4-2 Both SSDs

The run that utilized both SSDs is the only run done during this research that terminated because it used too many boxes. After the last iteration 1450336 boxes were still left. Because this run did not even converge to the box precision limits it is no surprise that none of the boxes can be considered trimmed. Reviewing the hull of the accelerations is also deemed pointless because of this. The bounds that were found are not as tight as the bounds found

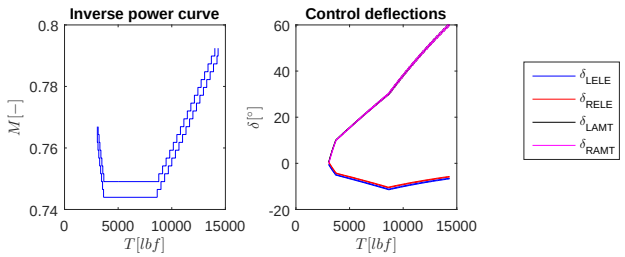

Figure 7-11: Trim enclosures as a function of thrust when using both AMTs at $\alpha = 5°$

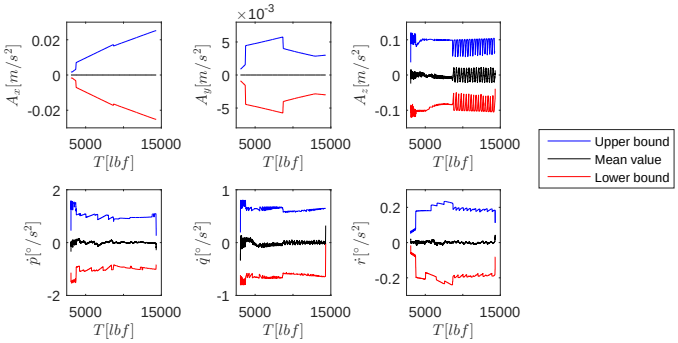

Figure 7-12: The hull of the leftover accelerations when using both AMTs

in previous runs, as can be seen in Figure 7-13. For the SSD deflections a profile starts to emerge when the code terminated. On the other hand do the elevon deflections still cover a very large area of their domain, so no conclusive notes can be made about this run. At the highest Mach numbers thrust values clip as the limit of what power the ICE model can deliver is reached. There are also some sections that cannot be trimmed using this control layout. Do keep in mind that the angle of attack is held constant at 5°. When comparing these results to the outcome of the run that used both AMTs it is interesting to see how two different devices intended for the same purpose can give such wildly different results. The AMTs can sweep through their entire range with a relatively small change in speed for the aircraft. The SSDs do not travel their full range, but in doing so the airspeed can be greatly increased.

7-4-3 LAMT and LSSD

If both the left AMT and the left SSD are used the algorithm finds a solution just a single box wide in mach number. If that is plotted it becomes completely unclear how the control effectors behave, so only the plots against thrust are given. This is done in Figure 7-14. From this plot it can be concluded that using the AMT is slightly more efficient. At the lowest thrust value the solution is similar to the one found with the run using just the left AMT at $\alpha = 5°$. At the highest thrust value the same can be said about the runs using just the left SSD. In between the two control effectors can be mixed to reach the desired effect. Since the case that uses only the SSD requires more thrust it can be concluded that the AMTs are more efficient, since the speed is the same for the full set of solutions in this run. The difference is only minor though. The enclosures found span only a very small area of the initial domain, less than 10^{-15} of the relative volume. The boxcount associated with this is also relatively small, peaking at 711 boxes. The accelerations in Figure 7-15 show that the choice of control effector does have an influence on the vertical body acceleration A_z. Using the AMT causes more of the hull of the accelerations to be below zero, while using the SSD pulls the average up to above zero.

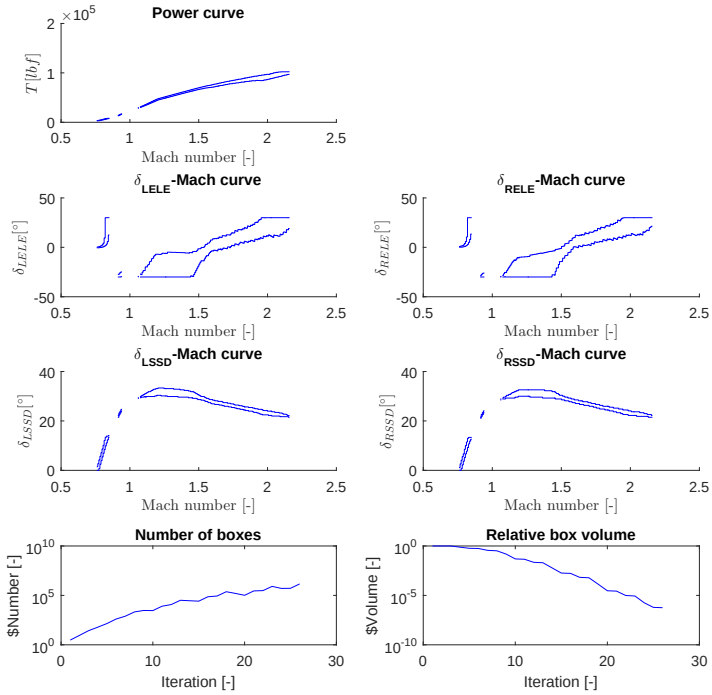

Figure 7-13: Trim enclosures as a function of Mach number when using both SSDs at $\alpha = 5°$

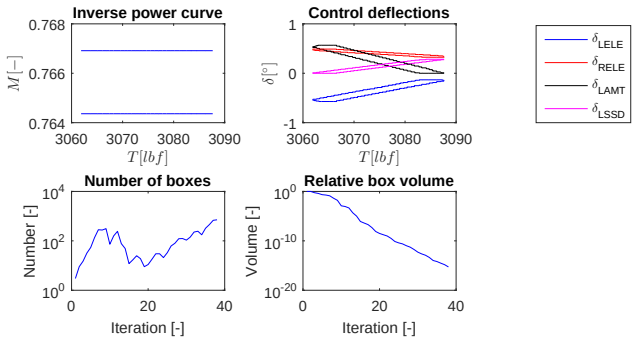

Figure 7-14: Trim enclosures when using the LAMT and the LSSD at $\alpha = 5°$

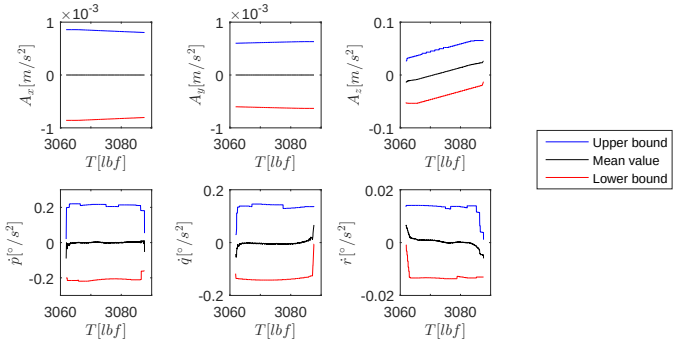

Figure 7-15: Leftover accelerations when using the LAMT and the LSSD at $\alpha = 5°$

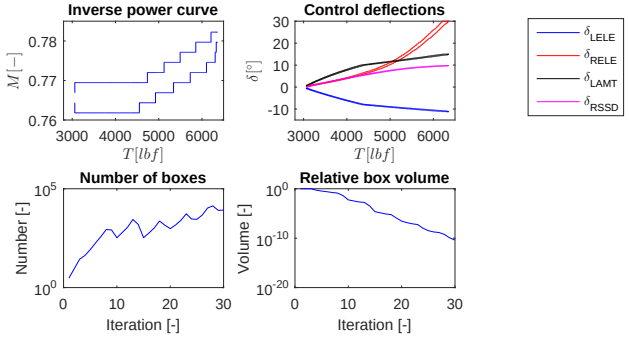

Figure 7-16: Trim enclosures as a function of thrust when using the LAMT and the RSSD at $\alpha = 5°$

7-4-4 LAMT and RSSD

All the previous runs that had an asymmetric layout only had it to balance the asymmetry in the model. This run is the first run to truly try an asymmetric configuration. The bounds on the deflections can be found in Figure 7-16. As with the previous run, the range in Mach number is quite small, so plots are made with respect to thrust. It can be seen that there can be highly asymmetric control deflections that still result in a trimmed state. Because the AMTs can only move trailing edge down and the SSD trailing edge up the elevons need to deflect in opposite directions to counter the moments on the aircraft. The limiting factor in this configuration is the right elevon. At the highest thrust setting in the plots it has reached its maximum deflection. This is the cause that there are not solutions at higher thrust settings and associated velocities. As can be seen in Figure 7-17 the amount of throttle influences how large the hull of the accelerations becomes. This is mainly notable in A_x, A_y and $\dot{r}$. That A_x becomes wider is caused by the width of the thrust intervals becoming wider at higher thrust values. A_y and $\dot{r}$ mainly become larger due to the bounds on the control deflectors becoming wider. This widening is best visible in the right elevon bounds.

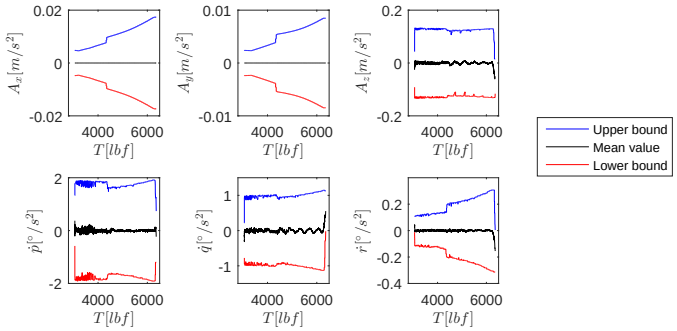

Figure 7-17: Leftover accelerations when using the LAMT and the RSSD at $\alpha = 5°$

Table 7-1: Demonstration of the dependency problem with B-splines

α	β	interval	points min max
[-0.1642, 0.2358]	[-0.2809, 0.0191]	[-2.5524, 0.7845]	[-0.0835, -0.0062]
[-0.1642, -0.0642]	[-0.2809, -0.1809]	[-0.2555, 0.0241]	[-0.0704, -0.0062]
[-0.1642, -0.1542]	[-0.2809, -0.2709]	[-0.0193, -0.0052]	[-0.0169, -0.0062]
[-0.1642, -0.1632]	[-0.2809, -0.2799]	[-0.0075, -0.0061]	[-0.0074, -0.0062]

7-5 Interval analysis and spline models

In Chapter 4 it was mentioned that interval analysis and spline models probably give rise to quite a significant dependency problem. To this extend a two variable fifth order spline model that calculates C_m as a function of α and β is evaluated here with interval arguments. The interval output is compared to an approximation of the tight inclusion interval. This approximation is calculated by evaluating a 100 by 100 grid of points equally spaced in the input domain. The interval arguments must be converted to barycentric coordinates first. If all data is on the same simplex as is the case in this example this conversion is quite straight forward. If multiple simplices were involved negative values of the barycentric coordinates would indicate that part of the interval lies on a different simplex. The negative barycentric coordinates indicate which simplex must be included in the evaluation as well. The spline model is evaluated multiple times with different widths of inputs. The results are shown in Table 7-1. It can clearly be seen that how wider the input interval is the more apparent the dependency problem becomes. In (De Visser et al., 2011) it is explained how the directional derivative of a spline function can be determined. This is still a function of the same type of basis functions. Therefore it can be expected that the dependency problem will also limit the capability of calculating tight bounds for the interval gradient of spline models, which can limit the usability of the gradient in contraction methods such as the interval Newton method.

7-6 Conclusion

This chapter has discussed results found during this research that were not treated in the paper. Although the results are promising they also show that more progress must be made

before the full potential of interval analysis is reached. Almost all the found results must be further refined before they could be considered truly trimmed. On the other hand it is possible to see trends of the trim curves in the current bounds for almost every case. It is also shown that using splines in combination with interval analysis is possible. It remains to be seen how effective it can be since interval dependency causes bounds to be wide if the input intervals are too large. A combination of using a data table model for the initial bounding and a spline model for final refinement could be a solution, making best use of both types of model.

Chapter 8

Conclusion and recommendations

This chapter will conclude the research done to find the trim points of the ICE model. After a summary the most important conclusions from this work will be discussed. Recommendations for further research will be done as well.

The objective of this research is to find the trim points of the ICE model. The ICE model has been developed by Lockheed Martin and does not feature a horizontal tail. Because of this there are three different control effectors that can provide yaw control. In addition to these three there are also other proven control effectors and multi-axis trust vectoring so that the model has a total of thirteen control effectors. This description answers research question 2.

Trim points are steady states of an aircraft. The aircraft will keep doing what it is doing. They are very useful for a multitude of reasons. They can for example be used as a safe set for flight envelope estimation or as design points for flight control systems. Finding the trim points comes down to finding the zeros of the dynamics of the aircraft model. Many different methods have been used to find the trim set. The method that has been used for this research is interval analysis. The reason for this choice is based on the fact that interval analysis is certain to find all possible solutions when solving the trim problem. This feature is especially powerful for the ICE model with its thirteen control effectors, which can result in a lot of different control deflections that would still be a trimmed condition. Interval analysis is also very good at dealing with constraints, a feature that has been used to keep the problem solvable with the available computing power. This section clarifies research questions 1, 3, 4 and 5.

To find trim points a zero finding algorithm has been implemented based on interval box consistency. The working principle of such an algorithm is to split the search domain in smaller pieces and trow away pieces that surely do not to contain any solutions. By doing this repeatedly interval enclosures of trim points can be found. This is the answer to research question 6. If the code is searching for single trim points the enclosures are in some cases so tight that the remaining accelerations would not be detectable by the human vestibular system in a motion simulator environment. Unfortunately this was not the case for all found points, that had remaining accelerations that were significantly larger, up to 0.3 m/s^2 and

$2°/s^2$. The true power of interval analysis is obvious when it is used to find continuous trim sets instead of points. In a single run it was capable of producing upper and lower bounds on the allowable control surface deflections for varying Mach numbers that are less than $1°$ apart. Despite these bounds being close the remaining accelerations were so large that the set could not be accepted as a trim set. The biggest problem is roll acceleration $\dot{p}$, which can be as high as $10°/s^2$. This means that research question 7 goes almost unanswered. But despite that the accelerations are still significant, the bounds on the control effectors are quite narrow. This means that there is a good starting point for further refinement.

Different runs have shown that when using redundant control effectors the ICE model can be trimmed at several velocities with the same angle of attack. This even holds for asymmetric cases. It was also one of these cases that could not converge more than it did because it would utilize more boxes than would be advisable for the computer running the calculations. The way the results are presented in this work is just one way that it can be done. Plotting everything as a function of the same variable is an easy way to see what combinations give what results. This answers research question 8. It should be kept in mind that it is not the only way to present trim sets.

To conclude it can be seen that interval analysis has a lot of potential as a trimming method. The ability of finding all possible trim states in one single run is a feature not found in conventional trim methods. Despite the promising results further research must be done to enable the finding of tighter solutions, so that the found results can be accepted as trim sets.

Possible improvements for the current method can be found in both the model and the solver. First of all the throughput of boxes is quite slow. The root of this problem lies in the data interpolation code. It is this piece of code that does not allow for use of a vectorized list of boxes. The trim routine runs in Matlab, a computational tool designed to be very efficient with matrix multiplication. If the interpolation functions are changed so that they allow for vectorization the computation time gains can be huge. Currently the algorithm uses a very basic box consistency based method to find solution enclosures. If the directional derivatives of the model are available interval Newton methods can be implemented. Instead of changing the current model it is also an option to use a B-spline model instead. This model is fully analytical over the entire domain, so it removes the need for interpolations. Spline based models also can provide directional derivatives without problems. Before this is pursued it must be made sure that the interval dependency that occurs when using intervals with spline models does not cause issues for the zero finding algorithm. The main focus should be to improve the efficiency of the current implementation by implementing any of the above proposals. The current code is capable of finding enclosures tight enough to be fully acceptable as trim set. The only issue is that in its current form it takes way too long.

Appendix A

Aerodynamic Equations

$$\begin{aligned} C_X = {} & C_{X_1}(\alpha, M) + C_{X_2}(\alpha, \beta, M) + C_{X_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) + C_{X_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & + C_{X_5}(\alpha, M, \delta_{llefi}) + C_{X_6}(\alpha, M, \delta_{rlefi}) + C_{X_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{X_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{X_9}(\alpha, \delta_{llefo}, \delta_{lamt}) + C_{X_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{X_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) + C_{X_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{X_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{X_{14}}(\alpha, \beta, \delta_{lamt}) + C_{X_{15}}(\alpha, \beta, \delta_{ramt}) + C_{X_{16}}(\alpha, \beta, \delta_{lssd}) + C_{X_{17}}(\alpha, \beta, \delta_{rssd}) \end{aligned} \tag{A-1}$$

$$\begin{aligned} C_Y = {} & C_{Y_1}(\alpha, M) + C_{Y_2}(\alpha, \beta, M) + C_{Y_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) - C_{Y_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & - C_{Y_5}(\alpha, M, \delta_{llefi}) + C_{Y_6}(\alpha, M, \delta_{rlefi}) - C_{Y_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{Y_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{Y_9}(\alpha, \delta_{llefo}, \delta_{lamt}) - C_{Y_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{Y_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) - C_{Y_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{Y_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{Y_{14}}(\alpha, \beta, \delta_{lamt}) - C_{Y_{15}}(\alpha, \beta, \delta_{ramt}) + C_{Y_{16}}(\alpha, \beta, \delta_{lssd}) - C_{Y_{17}}(\alpha, \beta, \delta_{rssd}) \end{aligned} \tag{A-2}$$

$$\begin{aligned} C_Z = {} & C_{Z_1}(\alpha, M) + C_{Z_2}(\alpha, \beta, M) + C_{Z_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) + C_{Z_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & + C_{Z_5}(\alpha, M, \delta_{llefi}) + C_{Z_6}(\alpha, M, \delta_{rlefi}) + C_{Z_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{Z_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{Z_9}(\alpha, \delta_{llefo}, \delta_{lamt}) + C_{Z_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{Z_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) + C_{Z_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{Z_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{Z_{14}}(\alpha, \beta, \delta_{lamt}) + C_{Z_{15}}(\alpha, \beta, \delta_{ramt}) + C_{Z_{16}}(\alpha, \beta, \delta_{lssd}) + C_{Z_{17}}(\alpha, \beta, \delta_{rssd}) \\ & + \frac{\bar{c}q}{2V} C_{Z_q}(\alpha, M) \end{aligned} \tag{A-3}$$

$$\begin{aligned} C_L = {} & C_{L_1}(\alpha, M) + C_{L_2}(\alpha, \beta, M) + C_{L_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) - C_{L_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & - C_{L_5}(\alpha, M, \delta_{llefi}) + C_{L_6}(\alpha, M, \delta_{rlefi}) - C_{L_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{L_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{L_9}(\alpha, \delta_{llefo}, \delta_{lamt}) - C_{L_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{L_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) - C_{L_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{L_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{L_{14}}(\alpha, \beta, \delta_{lamt}) - C_{L_{15}}(\alpha, \beta, \delta_{ramt}) + C_{L_{16}}(\alpha, \beta, \delta_{lssd}) - C_{L_{17}}(\alpha, \beta, \delta_{rssd}) \\ & + \frac{bp}{2V} C_{L_p}(\alpha, M) + \frac{br}{2V} C_{L_r}(\alpha, M) \end{aligned} \tag{A-4}$$

$$\begin{aligned} C_M = {} & C_{M_1}(\alpha, M) + C_{M_2}(\alpha, \beta, M) + C_{M_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) + C_{M_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & + C_{M_5}(\alpha, M, \delta_{llefi}) + C_{M_6}(\alpha, M, \delta_{rlefi}) + C_{M_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{M_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{M_9}(\alpha, \delta_{llefo}, \delta_{lamt}) + C_{M_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{M_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) + C_{M_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{M_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{M_{14}}(\alpha, \beta, \delta_{lamt}) + C_{M_{15}}(\alpha, \beta, \delta_{ramt}) + C_{M_{16}}(\alpha, \beta, \delta_{lssd}) + C_{M_{17}}(\alpha, \beta, \delta_{rssd}) \\ & + \frac{\bar{c}q}{2V} C_{M_q}(\alpha, M) \end{aligned} \tag{A-5}$$

$$\begin{aligned} C_N = {} & C_{N_1}(\alpha, M) + C_{N_2}(\alpha, \beta, M) + C_{N_3}(\alpha, M, \delta_{lele}, \delta_{lssd}) - C_{N_4}(\alpha, M, \delta_{rele}, \delta_{rssd}) \\ & - C_{N_5}(\alpha, M, \delta_{llefi}) + C_{N_6}(\alpha, M, \delta_{rlefi}) - C_{N_7}(\alpha, \beta, M, \delta_{llefi}, \delta_{llefo}) \\ & + C_{N_8}(\alpha, \beta, M, \delta_{rlefi}, \delta_{rlefo}) + C_{N_9}(\alpha, \delta_{llefo}, \delta_{lamt}) - C_{N_{10}}(\alpha, \delta_{rlefo}, \delta_{ramt}) \\ & + C_{N_{11}}(\alpha, \delta_{lamt}, \delta_{lele}) - C_{N_{12}}(\alpha, \delta_{ramt}, \delta_{rele}) + C_{N_{13}}(\alpha, M, \delta_{pf}, \delta_{lssd}, \delta_{rssd}) \\ & + C_{N_{14}}(\alpha, \beta, \delta_{lamt}) - C_{N_{15}}(\alpha, \beta, \delta_{ramt}) + C_{N_{16}}(\alpha, \beta, \delta_{lssd}) - C_{N_{17}}(\alpha, \beta, \delta_{rssd}) \\ & + \frac{bp}{2V} C_{N_p}(\alpha, M) + \frac{br}{2V} C_{N_r}(\alpha, M) \end{aligned} \tag{A-6}$$

Bibliography

De Marco, a., Duke, E. L., & Berndt, J. S. (2007). A General Solution to the Aircraft Trim Problem. *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*(AUGUST 2007), 1–40.

De Visser, C., Chu, Q., & Mulder, J. (2009). A new approach to linear regression with multivariate splines. *Automatica*(45), 2903-2909.

De Visser, C., Chu, Q., & Mulder, J. (2011). Differential constraints for bounded recursive identification with multivariate splines. *Automatica*(47), 2059-2066.

Dorsett, K. M., Fears, S. P., & Houlden, H. P. (1996). *Innovative Control Effectors (ICE)* (Tech. Rep.). Wright-Patterson Air Force Base.

Dorsett, K. M., Fears, S. P., & Houlden, H. P. (1997). *Innovative Control Effectors (ICE) Phase II* (Tech. Rep.). Wright-Patterson Air Force Base.

Elgersma, M. R., & Morton, B. G. (2000). Nonlinear Six-Degree-of-Freedom Aircraft Trim. *Journal of Guidance, Control, and Dynamics*, *23*(2), 305–311.

Georgy, M., & Evans, D. (2015, November). *Investigators '90 percent sure' bomb downed russian plane.* Available from `http://www.reuters.com` (Online; posted 8-November-2015)

Gill, P. E., Murray, W., & Saunders, M. A. (2002). SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Journal on Optimization*, *12*(4), 979–1006. Available from `http://epubs.siam.org/doi/abs/10.1137/S1052623499350013`

Hargreaves, G. I. (2002). *Interval Analysis in Matlab* (Tech. Rep. No. 416). Manchester: Manchester Centre for Computational Mathematics.

ICAO. (2016). *A Coordinated, Risk-based Approach to Improving Global Aviation Safety* (Tech. Rep.). Montreal: International Civil Aviation Organization. Available from `papers2://publication/uuid/F26C61BC-E3AF-46C6-AC70-F25C47ABC147`

Jaulin, L., Kieffer, M., Didrit, O., & Walter, E. (2001). *Applied interval analysis.* Springer-Verlag.

Juliana, S. (2006). *Re-entry flight clearance.* Delft University of Technology.

Kraemer, W. (2006). Generalized Intervals and the Dependency Problem. *Proceedings in Applied Mathematics and Mechanics*, *684*, 683–684.

Kwatny, H. G., Dongmo, J.-E. T., Chang, B.-C., Bajpai, G., Yasar, M., & Belcastro, C.

(2013). Nonlinear Analysis of Aircraft Loss of Control. *Journal of Guidance, Control, and Dynamics*, *36*(1).

Moore, E. R., Kearfott, R. B., & Cloud, M. J. (2009). *Global optimization using interval analysis: Interval optimization for aerospace applications.* Society for Industrial and Applied Mathematics.

Moore, R. E., & Yang, C. T. (1959). *Interval analysis 1* (Tech. Rep.). Sunnyvale, California: Lockheed Aircraft Corporation Missiles and Space Division.

Neumaier, A. (2003). Taylor Forms - Use and Limits. *Reliable Computing*, *9*(February), 43–79.

Paranjape, A., Sinha, N. K., & Ananthkrishnan, N. (2008). Use of Bifurcation and Continuation Methods for Aircraft Trim and Stability Analysis A State-of the-Art. *Journal of Aerospace Sciences and Technologies*, *60*(2), 85–100.

Rump, S. (1999). INTLAB - INTerval LABoratory. In T. Csendes (Ed.), *Developments in Reliable Computing* (pp. 77–104). Dordrecht: Kluwer Academic Publishers. Available from `http://www.ti3.tuhh.de/rump/`

Schittkowski. (1986). NLPQL: A Fortran subroutine for solving contrained non-linear programming problems. *Annals of Operations Research*, *5*(11), 485–500.

Shankar, P. (2013). Characterization of Aircraft Trim Points Using Continuation Methods and Bifurcation Analysis. *AIAA Guidance, Navigation, and Control (GNC) Conference*, 1–12. Available from `http://arc.aiaa.org/doi/10.2514/6.2013-4924`

Tupy, J., & Zelinka, I. (2008). Evolutionary algorithms in aircraft trim optimization. In *Proceedings - international workshop on database and expert systems applications, dexa.*

Van Kampen, E. (2010). *Global Optimization using Interval Analysis: Interval Optimization for Aerospace Applications.* Delft University of Technology.

Van Kampen, E., Chu, Q., Mulder, J., & van Emden, M. (2007). Nonlinear Aircraft Trim Using Interval Analysis. *AIAA Guidance, Navigation and Control Conference and Exhibit*(August). Available from `http://arc.aiaa.org/doi/10.2514/6.2007-6766`

Van Oort, E. R., Chu, Q. P., & Mulder, J. A. (2011). Maneuver Envelope Determination through Reachability Analysis. *Advances in Aerospace Guidance, Navigation and Control*, 91–102. Available from `http://link.springer.com/chapter/10.1007/978-3-642-19817-5_8`

Walster, G. W. (2004). *Global Optimization Second Edition , Revised and Expanded.* New York: Marcel Dekker, Inc.

Zink, P., Mavris, D., & Raveh, D. (2000). Maneuver trim optimization techniques for Active Aeroelastic Wings. *41st Structures, Structural Dynamics, and Materials Conference and Exhibit*. Available from `http://arc.aiaa.org/doi/10.2514/6.2000-1330`