University of Cape Town
Department of Mechanical Engineering

University of Cape Town

The Effects of Atmospheric Turbulence on Fuel Consumption in Extended Formation Flight

Drewan S. Sanders
Supervisor: C. Redelinghuys

Submitted to the University of Cape Town in partial fulfilment for the degree of Master of Science in Mechanical Engineering

February 2014

The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

University of Cape Town

Abstract

Extended formation flight (streamwise separations of between 10 to 40 spans), has been recently proposed as a method for reducing the induced drag of commercial aircraft. However, induced drag savings are not necessarily directly indicative of fuel savings. In a realistic environment, atmospheric turbulence will continuously perturb the formation's aircraft and their wakes. As a result, each aircraft in the formation will experience fluctuations in aerodynamic loads. For an aircraft to maintain accurately its position within a formation, it must continually adjust its throttle setting. This dynamic throttling may result in inefficient engine operation, thereby detracting from the reductions in induced drag. In this work, a high-fidelity transient engine model, representative of a typical commercial high-bypass turbofan engine, has been incorporated within a simple twin-aircraft formation flight simulator. The aerodynamic interactions between aircraft were modelled using a horseshoe vortex method, specially adapted for extended formations. The aircraft were constrained to longitudinal motion, with altitude fixed. This created a two degree of freedom formation model that is analogous to wind tunnel experimentation. A simple proportional gain controller was used to manipulate the throttle settings, in an attempt to maintain the trail aircraft's position relative to the leader, in a turbulent atmosphere. It was found that a fuel saving of approximately 25% may be achieved at a practical lateral separation of 1 span, corresponding to a stream-wise separation of 20 ± 0.3 spans, in moderate turbulence levels.

Acknowledgements

Firstly, I would like to express my sincere gratitude toward my supervisor and mentor, Prof. Redelinghuys, for his guidance in completing this research. His passion and enthusiasm for aviation was infectious, and I thoroughly enjoyed working with him.

I am very fortunate to have had the financial support of Airbus and the National Aerospace Center, who made this research project possible. During this time, I had the incredible opportunity to interact with Airbus' Andy Williams and Ian Marr, who's input and advice was invaluable.

I would also like to acknowledge Prof. Saravanamuttoo of Carlton University, who so kindly, and without hesitation, provided much needed information regarding the engine modelling in this work. Thank you ever so much.

I would like to thank my friends and colleagues; Nick Bizinos, Jordan Adams, William Tipping-Woods and Roberto Gomes, for the numerous discussions, input and ideas that cumulatively aided the completion of this work. A special thanks goes to Jordan Adams, who was always very patient in answering any queries I had. To the rest of the students, with whom I had pleasure of sharing a workspace with, thank you for all the good times and laughter.

I would like to dedicate this dissertation to my parents, who have sacrificed so much to get me to this point. Their unwavering love, patience, understanding and support during this entire process, has been absolutely incredible. I am forever indebted to them. I also wish to express a special thanks to my sister, who has continued to inspire me, with her incredible drive, determination and loving nature.

My final thanks goes out to Eleni Stenzel, who became my best friend during this time. I cannot thank her enough for the continual emotional support and encouragement she gave so selflessly, throughout the duration of this work. It is greatly appreciated.

There have been so many individuals who have contributed to this work in some way. Thank you all, it means so much to me.

Drewan S. Sanders
February, 2014

Contents

List of Figures

List of Tables

Nomenclature

Please note that symbols represented by Greek letters are listed first, and the remaining symbols listed in alphabetical order.

Formation Modelling Symbols

α	Angle of attack	
$\bar{q}$	Dynamic pressure	(Pa)
Δx	Longitudinal separation between aircraft	(m)
Δy	Lateral separation between aircraft	(m)
Δz	Vertical separation between aircraft	(m)
δ_e	Elevator deflection angle	(rad)
η	Spanwise separation $\frac{\Delta y}{b}$	$(span)$
Γ	Vortex Strength (Circulation)	(m^2/s)
μ	Viscous core radius $\frac{r_c}{b}$	$(span)$
ω	Temporal frequency	
Φ	Atmospheric turbulence spectra	
ρ	Density	(kg/m^3)
σ	Atmospheric turbulence rms intensity	(m/s)
σ	Formation influence factor	
θ	Pitch attitude (3DOF equations of motion	
ε	Downwash angle at tailplane	(rad)
ξ	Streamwise separation $\frac{\Delta x}{b}$	$(span)$

ζ	Vertical separation $\frac{\Delta z}{b}$	$(span)$
a_0	Tailplane zero incidence lift coefficient	$\left(\text{rad}^{-1}\right)$
a_1	Tailplane lift curve slope	$\left(\text{rad}^{-1}\right)$
AR	Aspect Ratio	$(-)$
b	Reference wing span	(m)
b_v	Span of bound vortex	(m)
$\bar{c}$	Mean aerodynamic chord	(m)
c	Aerodynamic chord	(m)
C_D	Coefficient of drag	$(-)$
C_L	Coefficient of lift	$(-)$
C_l	Lift coefficient per unit span	
C_m	Coefficient of pitching moment	$(-)$
C_{D_α}	$\frac{\partial C_D}{\partial \alpha}$	$\left(rad^{-1}\right)$
$C_{D_{\delta_e}}$	$\frac{\partial C_D}{\partial \delta_e}$	$\left(rad^{-1}\right)$
$C_{D_{\dot{\alpha}}}$	$\frac{2u}{\bar{c}}\frac{\partial C_D}{\partial \dot{\alpha}}$	$(-)$
C_{D_M}	$\frac{\partial C_D}{\partial M}$	$(-)$
C_{D_q}	$\frac{2u}{\bar{c}}\frac{\partial C_D}{\partial q}$	$(-)$
C_{l_α}	Lift Curve	$\left(\text{rad}^{-1}\right)$
C_{l_α}	Lift slope	
C_{L_α}	$\frac{\partial C_L}{\partial \alpha}$	$\left(rad^{-1}\right)$
$C_{L_{\delta_e}}$	$\frac{\partial C_L}{\partial \delta_e}$	$\left(rad^{-1}\right)$
$C_{L_{\dot{\alpha}}}$	$\frac{2u}{\bar{c}}\frac{\partial C_L}{\partial \dot{\alpha}}$	$(-)$
C_{L_M}	$\frac{\partial C_L}{\partial M}$	$(-)$
C_{L_q}	$\frac{2u}{\bar{c}}\frac{\partial C_L}{\partial q}$	$(-)$
C_{m_α}	$\frac{\partial C_m}{\partial \alpha}$	$\left(rad^{-1}\right)$
$C_{m_{\delta_e}}$	$\frac{\partial C_m}{\partial \delta_e}$	$\left(rad^{-1}\right)$

$C_{m_{\dot{\alpha}}}$	$\frac{2u}{\bar{c}}\frac{\partial C_m}{\partial \dot{\alpha}}$	$(-)$
C_{m_M}	$\frac{\partial C_m}{\partial M}$	$(-)$
C_{m_q}	$\frac{2u}{\bar{c}}\frac{\partial C_m}{\partial q}$	$(-)$
D	Drag	(N)
G_f	Fuel controller gain	
H	Altitude	(m)
h	Perpendicular distance to vortex filament	(m)
I_y	Mass moment of inertia (pitching)	$\left(kg.m^2\right)$
i_{th}	Upward inclination of thrust line from FRL	(deg)
L	Atmospheric turbulence scale length	(m)
L	Lift	(N)
l_{th}	Perpendicular distance from c.g. to thrust line	(m)
l_{tp}	Perpendicular distance from c.g. to tailplane c.p.	(m)
M	Mach number	$(-)$
m	Mass	(kg)
$\overset{\curvearrowright}{M}$	Moment	$(N.m)$
p	Roll rate	$(^{\text{rad}}/_s)$
q	Pitch rate (positive nose up)	$(^{rad}/_s)$
r	Radius	(m)
r	Yaw rate	$(^{\text{rad}}/_s)$
r_c	Viscous core radius	(m)
S	Reference wing area	$\left(m^2\right)$
u	Velocity component in the $\hat{\boldsymbol{i}}$ direction	$(^m/_s)$
V	Velocity	$(^m/_s)$
v	Velocity component in the $\hat{\boldsymbol{j}}$ direction	$(^m/_s)$
$\bar{V}$	Volume Ratio	$(-)$

$\boldsymbol{V}$	Velocity vector	$(^m/_s)$
V_θ	Tangential Velocity	$(^m/_s)$
w	Velocity magnitude in direction $\hat{\boldsymbol{k}}$	$(^m/_s)$
z_{tp}	Vertical distance (body axis system) of tailplane from FRL	(m)
a.c.	Aerodynamic Center (referenced to chord $\bar{c}$)	$(-)$
c.g.	Center of gravity (referenced to chord $\bar{c}$)	$(-)$
c.p.	Center of pressure (referenced to chord $\bar{c}$)	$(-)$

Engine Modelling Symbols

β	Auxiliary coordinate system applied to component characteristic maps. Refer to Appendix D	
δ	Corrected pressure. Refer to Equation D.4.	$(ratio)$
η	Efficiency	$(-)$
γ	Specific heats ratio	$(ratio)$
μ	Viscosity	$(^{Ns}/_{m^2})$
ϕ	Equivalence ratio	(ratio)
PR	Pressure ratio	$((ratio))$
ρ	Density	$(^{kg}/_{m^3})$
θ	Corrected temperature. Refer to Equation D.3. of Appendix D	$(ratio)$
φ	Heat transfer coefficient	$(^W/_{mK})$
A	Area	$\left(m^2\right)$
a	Speed of sound	$(^m/_s)$
C	Heat capacity	$(^J/_K)$
c_p	Constant pressure specific heat	$(^J/_{kg.K})$
c_v	Constant volume specific heat	$(^J/_{kg.K})$
D	Drag	(N)

E	Energy	(J)
E	Energy	(J)
e	Internal energy (mass specific)	$(^{J}/_{kg})$
F	Thrust	(N)
f	Mass specific body forces exerted on a fluid	$(^{N}/_{kg})$
$\bar{F}$	Specific thrust	(dimensionless)
$\boldsymbol{f}$	Vector of mass specific body forces exerted on a fluid in all directions	$(^{N}/_{kg})$
$\left(\frac{F}{A}\right)$	Fuel-air ratio	(ratio)
h	Enthalpy (mass specific)	$(^{J}/_{kg})$
I	Inertia	$\left(kg.m^2\right)$
$\hat{\boldsymbol{i}}$	Cartesian x-direction unit vector	
$\hat{\boldsymbol{j}}$	Cartesian y-direction unit vector	
K	Thermal conductivity	
$\hat{\boldsymbol{k}}$	Cartesian z-direction unit vector	
L	Length	(m)
M	Mach number	(dimensionless)
m	Mass	(kg)
$\dot{m}$	Mass flow rate	$(^{kg}/_{s})$
mf	Mole fraction	(ratio)
N	Spool speed	(rpm)
P	Pressure	(Pa)
p	Pressure (mass specific)	$(^{Pa}/_{kg})$
$\mathbb{P}$	Power	(W)
Q	Torque imbalance	$(N.m)$
q	Heat energy (mass specific)	$(^{J}/_{kg})$
$\dot{q}$	Heat transfer (mass specific)	$(^{J}/_{kg.s})$

R	Gas Constant	$(^{J}/_{kg.K})$
R_u	Universal gas constant	$(^{kJ}/_{kmol.K})$
s	Entropy (mass specific)	$(^{J}/_{kg})$
T	Temperature	(K)
t	Time	(s)
$\mathbb{T}$	Torque	$(N.m)$
TR	Temperature ratio	$((ratio))$
$TSFC$	Thrust specific fuel consumption	$(^{kg/s}/_{N})$
U	Velocity	$(^{m}/_{s})$
u	Velocity magnitude in direction $\hat{\boldsymbol{i}}$	$(^{m}/_{s})$
V	Volume	$\left(m^3\right)$
v	Velocity magnitude in direction $\hat{\boldsymbol{j}}$	$(^{m}/_{s})$
$\boldsymbol{V}$	Velocity vector	$(^{m}/_{s})$
W	Work	(J)
w	Velocity magnitude in direction $\hat{\boldsymbol{k}}$	$(^{m}/_{s})$
MW	Molecular weight	$(^{kg}/_{kmol})$
υ	Volume (mass specific)	$(^{m^3}/_{kg})$

Subscripts

∞	Free-stream condition
∞	Free-stream
eff	Effective
B	Body axis system
b	Combustion (burner)
BH	Burnham Hallock vortex profile
C	Corrected parameter. See Appendix D
c	Compressor

d	Atmosphere
e	Exit conditions
est	Estimate
F	Aircraft in formation
f	Fuel
g	Atmospheric turbulence gust
g	gas
H	Helmholtz vortex profile
HP	High pressure
I	Aircraft in isolation
i	Induced
i	Inlet conditions
IN	Inertial axis system
j	Lead aircraft
jk	Effect of lead aircraft j on trail aircraft k
k	Trail aircraft
LP	Low pressure
mix	Mixture
mol	Mole specific parameter
n	Nozzle
p	Constant Pressure
s	Isentropic
$stoich$	Stoichiometric
STP	Standard atmospheric conditions
t	Total or stagnation thermodynamic property (if absent static)
t	turbine
th	Thermal

th	Thrust
tp	Tailplane
des	Desired
m	Mechanical (Efficiency)
prod	Combustion Products
reac	Combustion Reactants
trim	Trim conditions

Abbreviations

AFF	Autonomous Formation Flight
AFT	Adiabatic Flame Temperature
BPR	Bypass ratio ($\dot{m}_{bypass}/\dot{m}_{core}$)
DOF	Degree of Freedom
DP	Design Point
FC	Flight Condition
FRL	Fuselage Reference Line: Longitudinal axis through aircraft in body reference frame
GPS	Global Positioning System
GSP	Gas-turbine Simulation Software
GUI	Graphical User Interface
HPC	High pressure compressor
HPT	High pressure turbine
HUD	Head Up Display
ICV	Inter-Component Volume
IPC	Intermediate pressure compressor
ISA	International Standard Atmosphere
LPCC	Low pressure compressor (fan core)

LPCD	Low pressure compressor (fan bypass)
LPT	Low pressure turbine

Please note that this nomenclature does not apply to symbols used in code presented in this work. The definitions of symbols used in each piece of code have been included within the preamble to that code. In addition, where symbols have been reused, the reader is kindly requested to refer to the context in which that particular symbol has been used

Chapter 1

Introduction

1.1 Background

The International Air Transport Association's (IATA) 2013 Annual Review revealed that the global aviation industry currently supports (directly and indirectly) 57 million jobs, transports up to 3 billion civilian passengers annually, and accounts for approximately 3.5% ($ 2.2 trillion) of the global GDP [1]. IATA highlighted that air connectivity is the key to economic growth; promoting global trade networks, tourism and business/social mobility. Despite the recent difficult global economic climate, the number of air travel passengers grew by 5.3% (between 2012-2013) and it has been predicted that 2016 will see an increase in 831 million passengers per annum compared to those recorded in 2011.

However, the sustainable growth of the aviation industry faces many challenges including; safety, air navigation capacity and efficiency, security, economic development and environmental concerns. Of the environmental concerns, IATA claims that the aviation industry is currently responsible for 2% of the world's CO_2 emissions. To reduce this contribution, IATA has set the international aviation community goals of halving CO_2 emissions by the year 2050 (compared to that of 2005), and reducing fuel consumption by 1.5% annually up till 2020.

As a result of these ambitious goals, the relevant policy makers, such as the International Civil Aviation Organisation (ICAO), have begun to impose more stringent emissions and fuel consumption regulations on airline companies [2]. In-turn airline companies are looking toward engine and aircraft manufacturers for improved technology to adhere to such regulations. A more comprehensive look at the effect of such regulations was discussed by Brueckner and Zhang [3] and more recently by Romli and Kamaruddin [4].

IATA has envisioned that such optimistic goals may be achieved through improvements in aircraft technology, using biofuels, finding more intelligent ways to fly and from supportive economic structures.

Over millions of years of evolution, animals have slowly perfected flight, and has thus found extremely intelligent methods of air travel. Therefore, it makes sense to look toward nature for inspiration and ideas on how to improve our own flight. The commercial aviation industry is focussed on finding more efficient ways to fly further. Therefore, we can look at the mechanisms which nature has developed to save energy and increase flight range. The primary example of this is the use of formation flight by migratory birds.

Various studies [5–10] have shown that birds are able to achieve large savings in energy through surfing the wake of other birds. It was then obvious to investigate whether formation flight could be applied to aircraft in a similar way to achieve similar benefits.

Investigations have confirmed that aircraft are able to achieve dramatic reductions in induced drag when flying in close-proximity formations. Blake and Gingras [11] managed a 25% reduction in the induced drag of two delta wings in formation during wind tunnel tests. Furthermore, a fuel saving of just over 18% was achieved by NASA's Autonomous Formation Flight Project during flight tests conducted on two F/A-18's flying in formation [12].

However, the commercial application of close-proximity formation flight is not feasible, mainly due to safety concerns. Ning *et.al.* [13] have proposed extended formation flight (i.e. stream-wise separation of around 20-40 spans between aircraft) as a safer alternative, and have demonstrated, analytically, induced drag reductions in the region of 30-40%.

Nevertheless, reductions in induced drag are not necessarily indicative of fuel savings. In atmospheric turbulence, engine throttle settings will have to be continuously adjusted to maintain each aircraft's position within the formation to achieve optimum aerodynamic benefit. This will in-turn decrease the efficiency at which the engine operates. Therefore, the effects of dynamic throttling must be accounted for when evaluating the benefits of extended formation flight in realistic environment.

1.2 Research Objectives

The objective of this work, is to provide better insight into the potential fuel savings achievable through the implementation of extended formation flight. To that end, it was the aim of this work to investigate longitudinal formation dynamics, in a realistic atmosphere, that result in inefficient use of the formation.

To achieve the objective above, it is necessary to develop a number of models, that when combined, will allow for the simulation of formation dynamics in a realistic atmosphere. The first model to be developed, is that which calculates the aerodynamic interactions between aircraft within a for-

mation. Next, it is necessary to model the atmospheric turbulence in which the formation will fly. Subsequently, each aircraft (representing a typical large commercial airliner) must be modelled in terms of its equations of motion, which are influenced by the both the formation and atmospheric turbulence. These components are required to model the formation aerodynamically.

Next, a transient engine model was required. As mentioned earlier in this introduction, the aircraft will have to constantly adjust their throttle settings in order to maintain their beneficial positions within the formation. A transient engine model, typical of that used by a large commercial airliner, is required to provide accurate time responses of the engine to dynamic throttling, as this will effect the dynamic response of the aircraft within the formation. In addition, such an engine model is crucial in accurately predicting the fuel consumption required to achieve the dynamic thrust demands of the aircraft. This prediction of fuel consumption formed the key objective of this work.

Finally, a simple control system must be developed to manage the aircraft's position within a formation, by manipulating the throttle settings of the aircraft's engines. This control system will tie together the previous models mentioned, thereby forming a complete model of the formation dynamics.

The final formation dynamics model is to be "flown" in various levels of turbulence to establish the kind of fuel savings that may be achieved. In so doing, it is also the aim of this work to establish station-keeping tolerances which allow for optimum fuel savings.

1.3 Document Description

The second chapter, Chapter 2.2, introduces the concept of formation flight. This literature review first explores formation flight as used by birds in nature, by examining theoretical and experimental studies on the subject. Subsequently, the applications of close-proximity formation flight to aircraft is discussed, with reference to both theoretical and experimental studies. Next, the recent idea of extended formation flight for commercial application is considered. Finally, the concern of dynamic throttling is presented, along with some examples found in literature.

Chapter 3 follows the development of the formation dynamics model produced in this work. First, the aerodynamic interactions between aircraft are discussed. This discussion is followed by a description of atmospheric turbulence and its effects on the aerodynamics of the formation and individual aircraft. Finally, the simplified dynamic representation of the formation is presented, along with a control system to manage the formation geometry.

The following chapter, Chapter 4, introduces jet engines and their use in

propulsion. This literature review covers jet engines, from their conception to modern day form. The various methods of simulating engine dynamics are also presented, along with existing software. Next, in Chapter 5, the engine model developed for the purposes of this work, is described in detail. This chapter also includes extensive validation of the engine model developed.

Finally, the engine model is incorporated within the formation dynamics model. This model is then used to explore the possible fuel savings in different formation flight regimes. The results of which are presented and discussed in Chapter 6. Finally, conclusions and recommendation, based on the results produced, are voiced in Chapters 7 and 8.

Chapter 2

Formation Flight

Formation flight is not a new concept, however technological advancements have made new applications feasible. This chapter begins with a discussion in the basic principles of formation flight, and follows the progression of its research from the study of birds to possible commercial applications in the aviation industry.

2.1 Basic Principles of Formation Flight

Formation flight has long been identified as an energy saving mechanism, which is quite evident from basic aerodynamic principles. Downwash occurs directly behind a finite wing as a result of the momentum change in air which causes lift. Outside of its span, upwash occurs as a result of flow about the wing tip, as the air moves from a high to low pressure region. This causes a vortex to trail behind each wing tip as indicated in Figure 2.1.

If a trailing aircraft flies into the upwash of another, its lift vector is tilted forward, thus increasing and reducing its vertical and horizontal components respectively (see Figure 2.2). In doing so, a trailing aircraft may reduce power and pitch to maintain constant speed and altitude. The change in the horizontal component is much larger, and therefore decreasing propulsive power is of primary importance for maintaining position within a formation.

There are two primary methods used for calculating the change in induced drag of aircraft in a formation. The first method substitutes each aircraft with a single horseshoe vortex, although in some cases the main wing and tail plane are both replaced with a horseshoe vortex for better representation. In accordance with Prandtl's lifting line theory, the horseshoe vortex consists of a vortex filament which is bound within the span of the wing and two trailing filaments that extend infinitely down-stream (in accordance with Helmholtz's theorem) from the ends of this bound filament. Horseshoe vortices, representing the aircraft in a formation, will induce flow effects on one another based on their individual vortex strengths and three-

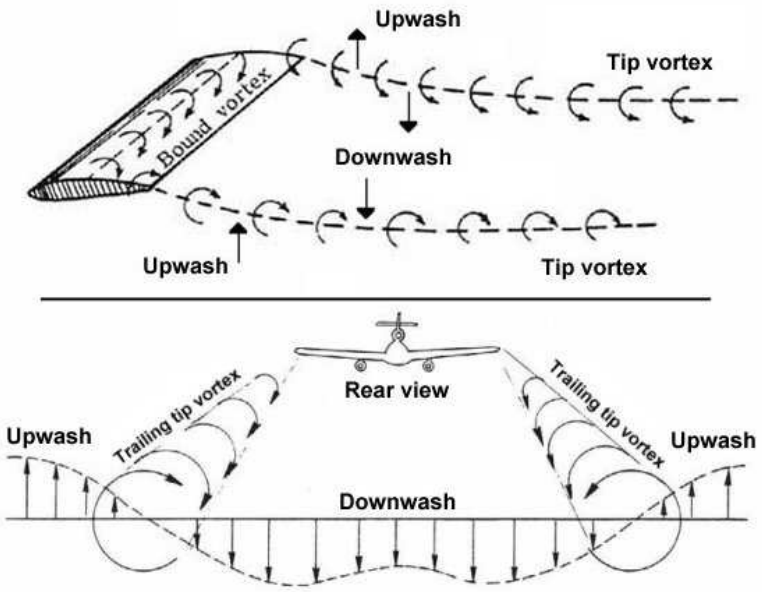


Figure 2.1: Two counter-rotating vortices trail behind the wingtips of an aircraft, where they induce a flow field with upwash and downwash regions. Reprinted from aerospaceweb.org [14]

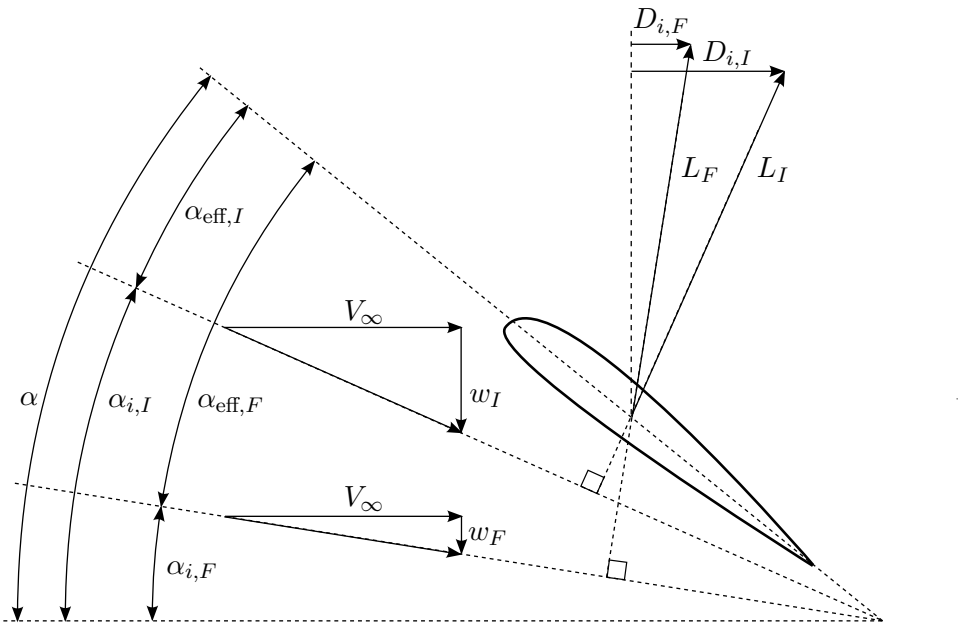


Figure 2.2: Lift vector of an airfoil in formation versus isolation.

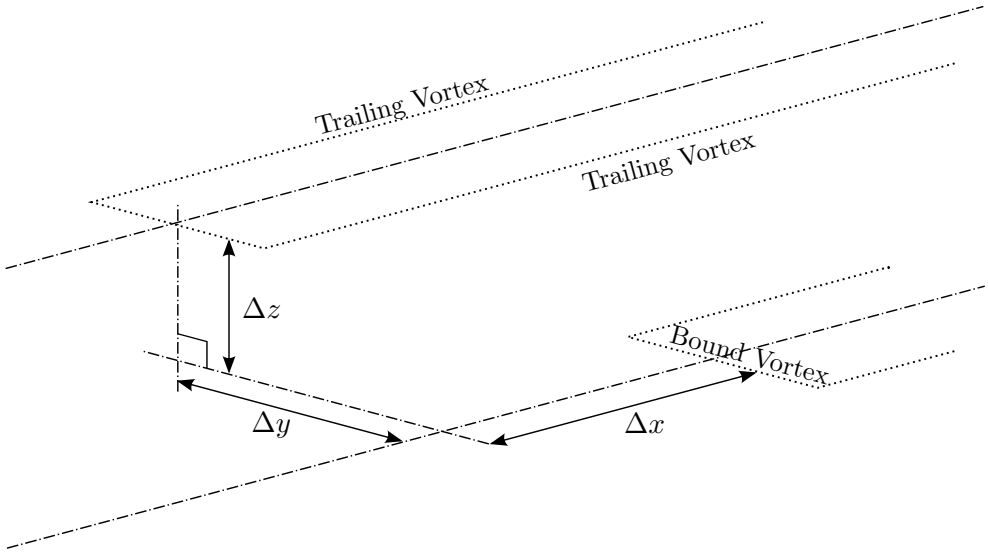


Figure 2.3: Horseshoe vortex schematic representing two aircraft in formation

dimensional positioning relative to one another. These induced effects are calculated in accordance with the Biot-Savart law, to ascertain lift distributions when in formation as opposed to isolation. The reader is referred to Anderson [15], for more details on these theorems and techniques. A simple horseshoe vortex representation of a two aircraft formation flight is illustrated in Figure 2.3, where each aircraft has been represented as a single horseshoe vortex.

The vortex lattice method is an extension of the horseshoe vortex method. Here, a wing is represented as a series of Prandtl's lifting lines placed incrementally along the chord. The lattice forms two vortex sheets, the first across the wing span and the second in the stream-wise direction. Combined these are referred to as a lifting surface, which is better for representing more complex wing geometries, such as the delta wings shown in Figure 2.4. Numerical methods are implemented to calculate the aerodynamic interactions between aircraft using the Biot-Savart Law, as before.

An important concept in formation flight is Munk's Stagger Theorem, which states that only the distribution of induced drag saving, and not its cumulative total, changes with variations in streamwise separation between aircraft. [16]. However, Munk's stagger theorem assumes inviscid and incompressible flow, and does not account for vortex decay and stability, for example, which detracts from savings with increased longitudinal separations. Nevertheless, this theorem becomes important when incorporating aircraft of different sizes and mass within a formation. This is because the distribution of induced drag saving changes along the formation. Therefore,

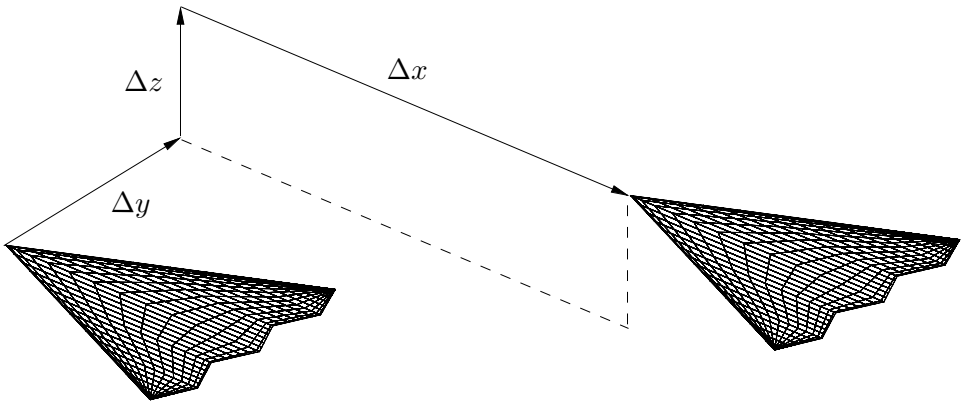


Figure 2.4: Vortex lattice schematic representing two delta wings in formation. Reprinted from Blake and Gingras [11]

the formation may be optimized by placing certain aircraft in certain positions within the formation. Heavier and bigger aircraft generate stronger trailing vortices, and therefore the total induced drag saving of the formation may be increased by optimizing the weight distribution across the formation. This has been discussed in detail by Blake and Multhopp [17].

In addition, there is a limit to the number of aircraft which may be included in a formation, predominantly due to formation dynamics. Nevertheless, it is now necessary to discuss the occurrence of this principle, and these topics will be revisited later on.

2.2 Birds in Formation Flight

Some of the first techniques for quantifying the aerodynamic benefit of formations, were developed to analyse formation flight in birds. In nature, flocks of birds have been observed to fly in many types of formations, particularly during long migrations. Avian formation flight is phenomena that has been perfected over millions of years. Initial explanations as to the purpose of such formations were discussed by Heppner [5]. The two main hypotheses presented, argued orientation for visual communication versus energy savings due to aerodynamic effects.

Badgerow [6] found evidence, through analysing formation geometries of geese, to support both claims, but concluded that energy saving purposes appeared to be of higher priority. This conclusion was based on the work by Lissaman and Shollenberger [7], who were among the first theoretically model the benefits of formation flight in birds. Hummel [8] supported this by demonstrating a significant induced drag reduction through numerical calculations using both horseshoe vortex and lifting surface techniques.

Nevertheless, Weimerskirch *et al.* [9] later provided concrete experimen-

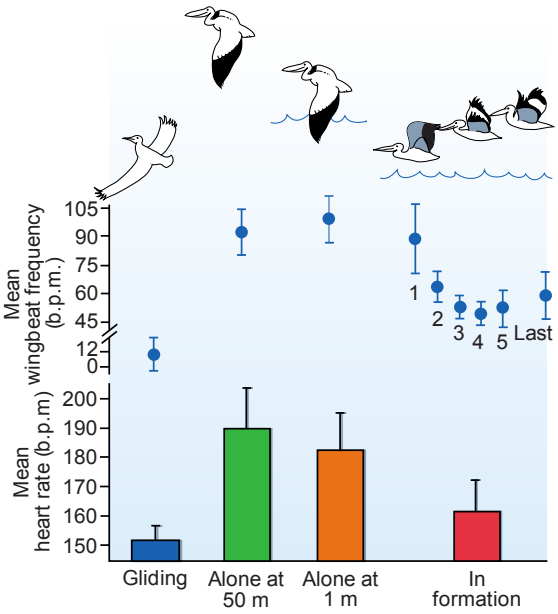


Figure 2.5: Variations in wing-beat efficiency and heart rate for birds flying in formation versus isolation. Reprinted from Weimerskirch *et al.* [9]

tal evidence of aerodynamic benefit, over visual communication, by recording the differences in heart rate of pelicans flying in formation versus isolation. These findings, summarised in Figure 2.5, indicated a reduction in heart rate of up to 14.5% and an overall energy saving of up to 3.4%, which correlate reasonably well with predicted savings of 2.4% calculated by Cutts *et al.* [10] for studies on geese.

In Figure 2.5, it is interesting to note the distribution of saving across a six bird formation. The birds in the middle of the formation have the lowest wing beat frequency, indicating that they benefit the most from the formation. This correlates well with theoretical studies on aircraft by Blake and Multhopp [17]. Through birds, nature has demonstrated the potential benefits of flying in formation. The question is then whether this same principle may be exploited by aircraft to achieve similar energy savings?

2.3 Aircraft in Close-Proximity Formations

Technological advancements in flight control systems sparked interest in the use of formations to increase the efficiency of manned flight. After further research contributions regarding bird formations, Hummel turned his attention toward applying formation flight to aircraft. In one of his more recent papers [18], Hummel compared theoretical predictions from horseshoe vortex and vortex lattice methods against flight test data. The

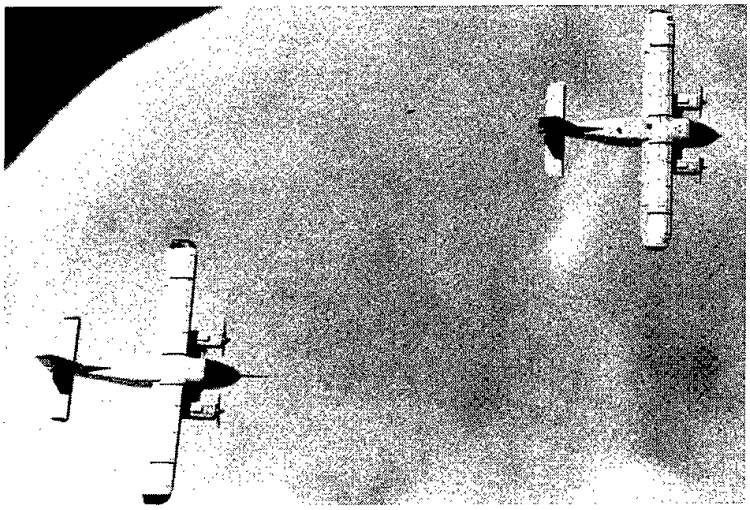


Figure 2.6: Two Dornier DO-28's flying in formation to record power reductions for comparison against theoretical calculations. Reprinted from [18].

flight tests, conducted on two Dornier DO-28's, demonstrated a 10% power reduction for a trailing aircraft, which correlated well to the 15% predicted theoretically. Figure 2.6 shows a picture of the DO-28's taken during the flight test.

Blake and Multhopp [17] furthered research of formations by considering the implications of different formation geometries, number of aircraft, varying aircraft sizes and propulsive limitations (as was touched on earlier). This analysis was performed using both horseshoe vortex and vortex lattice modelling techniques. An echelon formation was recommended as a safer formation geometry to that of the "V" or inverted "V" types. An optimum formation effect was achieved through distributing the weight across the formation elliptically, and rotation of the lead aircraft with a frequency proportional to the number of aircraft in the formation. The rotation of lead aircraft is necessary because of the varying fuel burn, which in-turn varies the weight of aircraft in the formation. It was also shown that increasing aircraft numbers in a formation furthers the benefit achieved, however as the formation exceeds about six aircraft, the change in benefit reduces significantly.

The final results of Blake and Multhopp's study, demonstrated that a 60% increase in range was possible for a five aircraft formation. However, this saving is highly dependent on vortex tracking. If the trailing vortices are not tracked within one tenth of a wingspan, 50% of the drag benefit may be lost. In realistic conditions, atmospheric turbulence will increase such tracking errors.

Later, Blake and Gingras [11] compared predictions, calculated from

Figure 2.7: NASA Autonomous Formation Flight Project FA18 Flight Tests. Reprinted from [19].

vortex lattice methods , against results from wind tunnel tests on two delta wings in formation, as per Figure 2.4. In these experiments a maximum induced drag reduction of 25%, compared to a theoretical 40%, was achieved.

NASA's Dryden Flight Research Center launched their *Autonomous Formation Flight Project* in the early 2000's. Close-proximity formation flight tests were performed on two F/A-18's to verify theoretical predictions [12, 19, 20]. The flight tests were able to demonstrate a 20% and 18% reduction in induced drag and fuel consumption respectively. A photograph taken during one of the tests, is depicted in Figure 2.7.

The NASA flight test results, illustrated that savings in fuel burn and induced drag are not directly proportional, and in fact the fuel savings are lower. This perhaps indicates that inefficiencies are introduced by the dynamic throttling of the engines required to maintain the aircraft's position within the formation, therefore motivating further investigations into formation flight with an emphasis on engine performance (which is the topic of this particular work).

Nonetheless, the theoretical and experimental work presented above, suggests that significant energy savings may be achieved through flying aircraft in formations, rather than in isolation. However, the follow-up question asks whether formation flight is appropriate for commercial applications, where such savings are required the most?

2.4 Commercial Application of Formations

The commercial application of formation flight has been explored because of many potential benefits. Over and above possible fuel savings, formation flight could serve to alleviate air traffic congestion and improve inter-aircraft communications, to name a few [21]. Some of the aspects of implementing formation flight commercially have also been explored macroscopically in terms of logistics and optimization [21–23].

Nevertheless, one of the biggest challenges of implementation are those concerning safety. Flying commercial aircraft in close proximity formations could be extremely hazardous, with a high risk of mid-air collisions. Therefore extended formations have been proposed as a safer alternative, by reducing risks of mid-air collisions, whilst still achieving significant fuel savings [13]. This idea is based on the persistent nature of aircraft contrails, which allow aircraft to follow one another at safer distances, whilst still benefiting from the upwash region of the trailing vortices. This idea coincides with Munk's Stagger Theorem [16], which states that the induced drag benefit of a formation is independent of variations in longitudinal separation.

However, in a realistic compressible and viscous environment, vortices decay, become unstable and are shifted about by atmospheric turbulence. Therefore, induced drag benefits will diminish with increasing longitudinal separation.

The first investigations into extended formations [13, 24], have revealed that aircraft may achieve substantial benefits for streamwise separations of up to 40 to 50 spans. Initial predictions revealed that 2 and 3 aircraft formations could, typically, achieve induced drag reductions in the order of 30 and 40 % respectively [13]. Further investigations were performed to investigate the effects of compressibility, by comparing formations flying at subsonic and transonic speeds, as well as the effects of aircraft trim [24]. This more recent work revealed that, for the trailing aircraft in a 2 aircraft formation, reductions in induced drag could reach values as high as 54 and 35% for subsonic and transonic speeds respectively. Highlighted in both works, was the concern of vortex tracking errors, particularly resulting from the influence of atmospheric turbulence on aircraft and wake dynamics. As a result it was concluded that formation flight would only be practically beneficial in low to moderate turbulence levels [13].

Although extended formations do tackle the issue of safety, it does unfortunately amplify tracking errors. Figure 2.8 illustrates tests results, from NASA's AFF project, which reveal the degradation of induced drag benefit with increasing longitudinal separations [12]. These results indicate a gradual decrease in aerodynamic benefit for an increase in streamwise separation between aircraft. However, these results do indicate that significant savings are still achievable at separations up to ten wingspans.

As with the study by Blake and Multhopp [17], Ning *et al.* [13] consid-

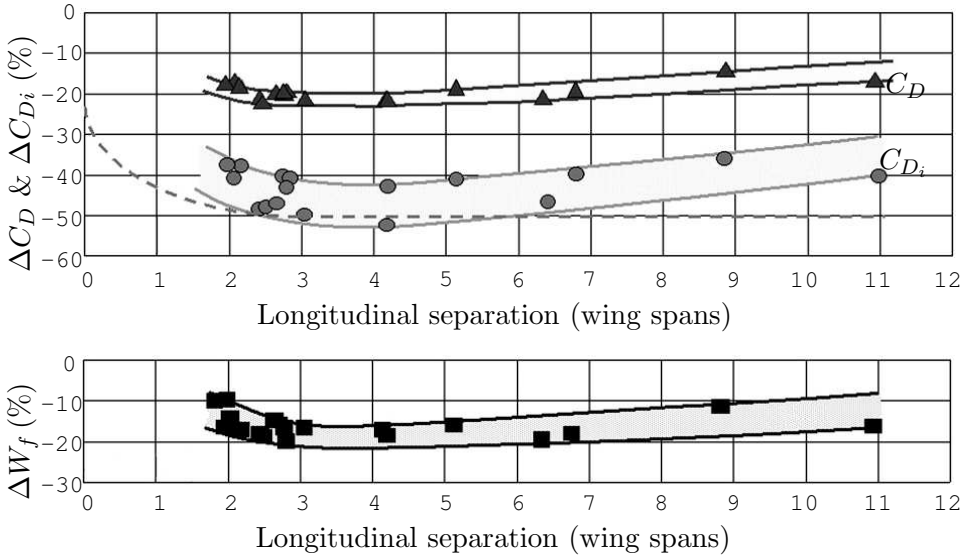


Figure 2.8: NASA AFF test results showing the effect of longitudinal separation on formation performance The dashed line represents the theoretical prediction, and the solid lines are curve fits to the experimental data. Reprinted from [12]

ered different formation configurations. The aim of these comparisons was to determine the influence of varying atmospheric turbulence and stratification on their performance. The three aircraft formation configurations tested were; the "V", inverted "V" and echelon, of which the inverted "V" generally achieved the least benefit. However, trim drag due to viscous and compressibility effects was not considered, and the inverted "V" would suffer the least in this regard due to a more symmetrical lift distribution on the trailing aircraft. It was also observed that the performance of the echelon formation decreased faster with increasing longitudinal separations. In the echelon formation, the interaction of vortices from the first two aircraft increases the uncertainty of the lateral and vertical separation required of the third aircraft for optimum benefit. Thus, the tracking error of the third aircraft will increase with longitudinal separation.

Overall, increasing the longitudinal separation results in greater uncertainty in the position of the trailing vortices which has an adverse effect on drag savings. In addition, an increase in turbulence causes a greater shift in vortex positioning and subsequently corresponds to a greater vortex tracking error and further decrease in drag savings. Therefore, further studies on the actual benefit achievable are required; in particular, those concerning the true effects of dynamic tracking errors. One concern is that regarding dynamic throttling of the engines (required for tracking vortices) which will detract from the aerodynamic benefits achieved in a formation.

Other aspects of extended formations have been investigated. The most recent published study on extended formations, has investigated the effects of formation flight, within atmospheric turbulence, on passenger comfort [25]. Investigations such as these are important, in order to analyse the use of extended formations from a practical point-of-view.

Nonetheless, the aim of this work is to analyse the use of extended formation flight from a fuel burn perspective.

2.5 Dynamic Throttling in Formation Flight

In the previous section, the literature indicated that vortex tracking error increases with longitudinal separation and atmospheric turbulence, which in-turn drastically reduces induced drag savings. In order for an aircraft achieve the most benefit in formation, it must constantly attempt to maintain an optimum relative vertical and lateral separation (formation performance is far less sensitive to variations in longitudinal separation, in accordance with Munk's stagger theorem). Nonetheless, to maintain altitude, airspeed and correct lateral positioning, the aircraft must constantly adjust its throttle setting. Dynamic throttling works against inertias within the engine, thus reducing its operating efficiency. In addition, the engine's efficiency is decreased further because it is unable to settle on the design cruise condition, as it would normally do for an aircraft flying in isolation. Therefore, there exists a trade-off between efficient engine operation and induced drag reduction from station-keeping precision. Ideally, a formation autopilot would be responsible for optimising this trade-off by managing engine throttle settings.

One of the first autonomous formation flight control systems was developed for the flight tests in Hummel's study [18] mentioned earlier. The trailing aircraft implemented adaptive extremum peak-seeking control, by using aileron deflection, to track the greatest region of upwash generated by the trailing vortex of the lead aircraft. This control system attempted to maintain a constant speed in the longitudinal direction, while purposely adjusting vertical and lateral separations to ascertain the position of highest power reduction. Figure 2.9 presents results for the longitudinal motion of the trailing aircraft, and indicates how the control system attempts to settle on the same altitude as the wake of the lead aircraft. More importantly, this figure depicts the fluctuations in thrust required to maintain a constant speed as the trailing aircraft moves in and out of the optimum aerodynamic region.

A more recent example of a formation flight autopilot, was that developed during NASA's AFF program. NASA's AFF project's aim was to design and test an autopilot that could be used to reduce fuel consumption for long range missions [12, 19, 20]. The three phase programme was aimed

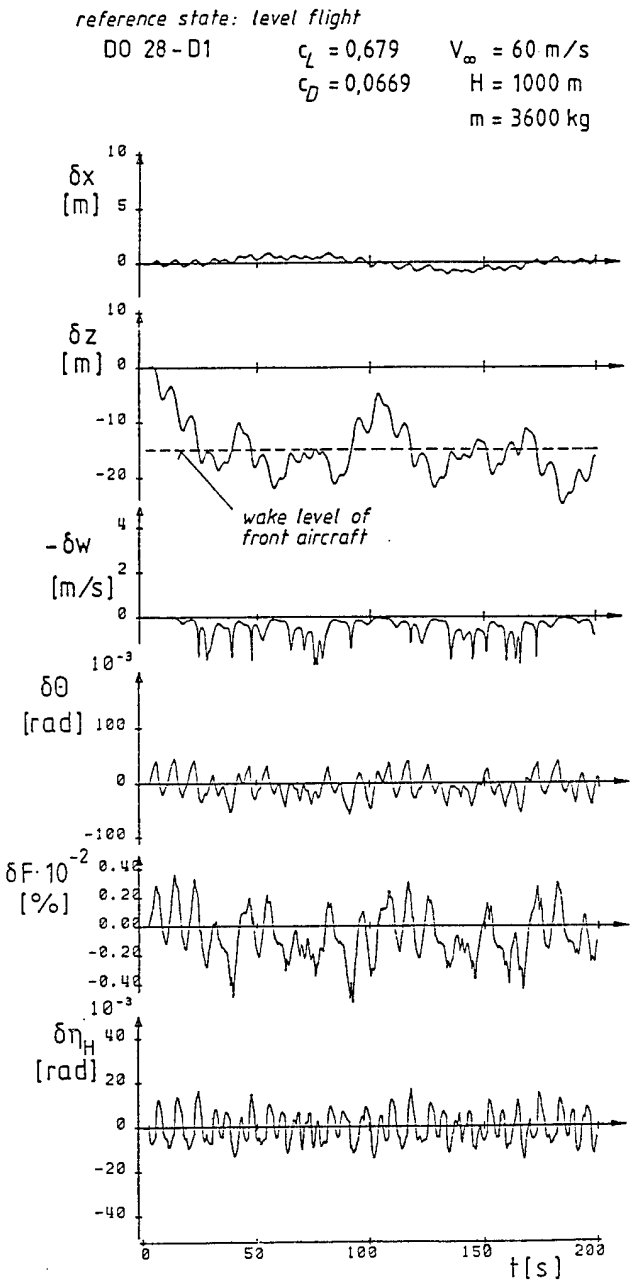


Figure 2.9: Longitudinal motion of trailing aircraft during flight test. δx, δz, δW, δF symbolize the longitudinal separation, vertical separation, downwash and thrust respectively. Reprinted from [18].

at:

- Demonstrating autonomous station-keeping precision
- Verifying theoretical results by mapping the effects of a lead aircraft's vortex on a trailing aircraft's performance
- Integrating the autopilot to establish whether a 10% fuel saving could be maintained in automated flight

Phase one of the AFF programme used GPS to calculate relative positions between aircraft, which was then conveyed to the pilot via the HUD. The HUD displayed the error between the actual and desired vertical and lateral separation between aircraft. The pilot could then make manual corrections to track the desired position, whilst the thrust and fuel flow were monitored by in-flight systems. A tracking accuracy of $2.5 \pm 2ft$ was achieved.

In phase two of the programme, an automatic throttle control was used to maintain the trail aircraft's position for different pre-programmed vertical and lateral separations. The lead aircraft was flown at constant speed and altitude for this test. The AFF programme was cancelled after this second phase due to budget constraints.

Nevertheless, Figure 2.10 shows a sample of the results obtained during the second phase of the AFF programme. During this test the pilot initially used manual control to keep formation. The automatic throttle control was applied later to smooth out the cyclic throttling. To end the test, the trailing aircraft slid out laterally from the influence of the lead aircraft's trailing vortices. The results in Figure 2.10 illustrate an oscillating reduction in drag for the trailing aircraft, accompanied by cyclic throttling required to maintain station within the formation. The fuel flow measurements indicate that the mean fuel consumption of the trailing aircraft is still less than that of the lead aircraft (19.2% less for the sample test shown), despite inefficient engine operation due to transients. However, it must be noted that the engines on the F/A-18's are better designed to handle transients than those used in civilian transportation. It can also be seen that the amplitude of fuel flow oscillations were drastically reduced by the automated throttle control. The fluctuations in Mach number on the lead aircraft indicate the presence of atmospheric turbulence, which appears to be amplified onto the trailing aircraft, by the wake of the lead aircraft.

Figure 2.8 indicated how little the induced drag saving diminished with increased longitudinal separation. This implies that, for extended formations atleast, a relatively low gain throttle controller may be used to gradually alter the throttle setting required to maintain station within a formation. This would therefore reduce the negative effects of dynamic throttling on engine performance. However, loose streamwise separation tolerances may be become impractical as the number of aircraft in the formation is increased.

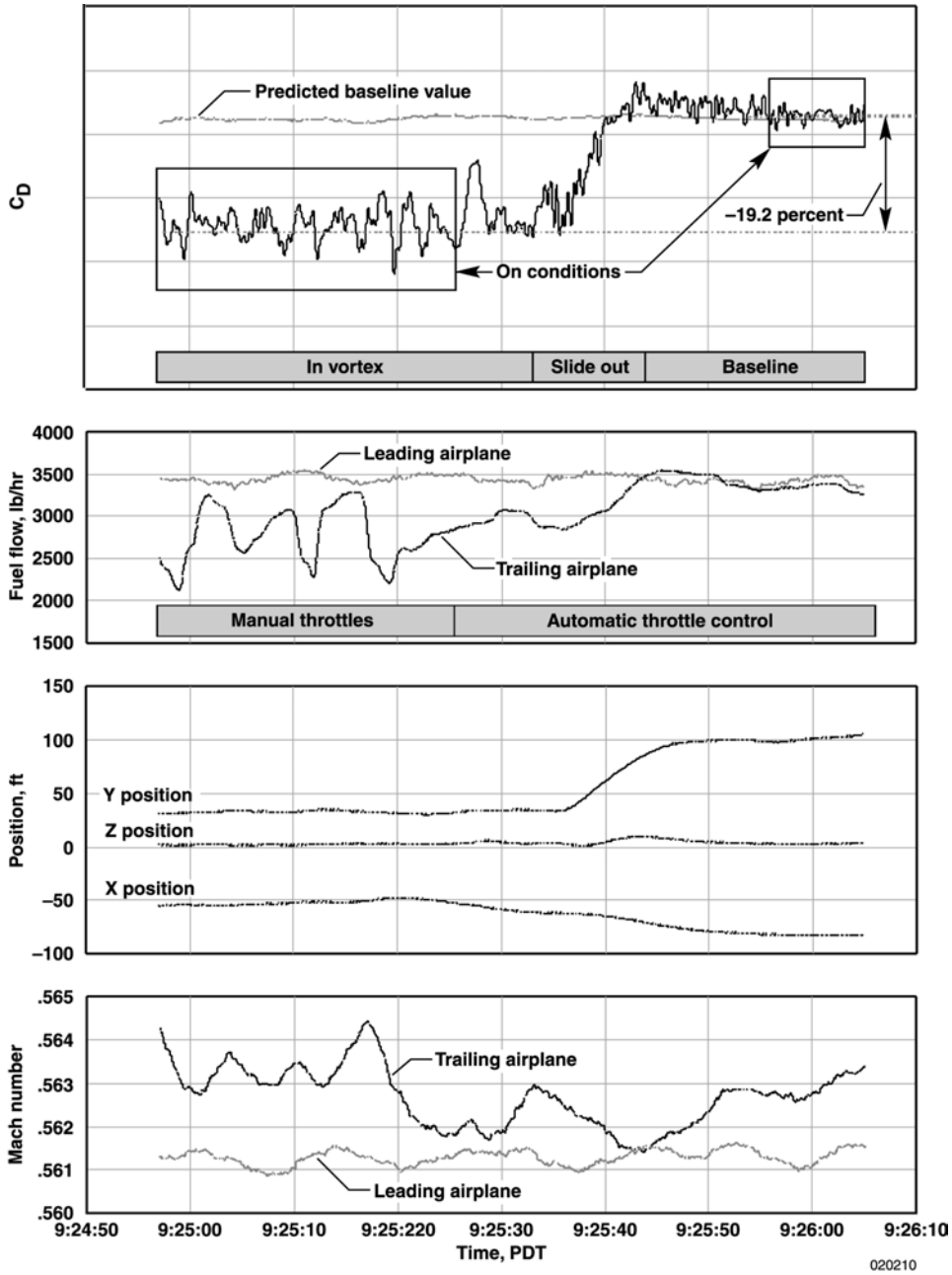


Figure 2.10: Sample of results from NASA's AFF programme. Reprinted from [19]

It certainly appears, from the literature presented above, that significant fuel savings are achievable despite any adverse effects of dynamic throttling. However, thus far the literature has only considered the effects of dynamic throttling applied to small aircraft in close proximity formations. It is, therefore, of value to consider how dynamic throttling would effect the efficiency of large commercial airliners in extended formations

2.6 Summary

This chapter first introduced the basic concept of formation flight. Here, it was explained how an aircraft may benefit aerodynamically by flying into the upwash generated by another aircraft. Some techniques, namely the horseshoe vortex and vortex lattice method, used for modelling this aerodynamic benefit were mentioned.

After discussing the basic principles of formation flight benefits, the chapter presented a discussion on how birds have used formations to save energy. This included both theoretical and experimental studies on the topic. Following the discussion on avian formations, it was discussed how aircraft may use bio-mimicry to achieve similar benefits. Literature, regarding theoretical and experimental investigations into close proximity aircraft formations, was presented. Of particular interest, was the flight tests performed on two FA-18's during NASA's AFF programme, which revealed promising fuel savings of up to 18%.

Next, the question was raised as to whether formation flight could be applied commercially to large passenger airliners? It was argued that the close proximity type of formation flight would be too dangerous for application to large airliners. However, a relatively new idea of extended formations was discussed. In this idea, aircraft follow one another at safer distances, whilst still receiving aerodynamic benefit.

The literature presented revealed, in theory, that enormous reductions in induced drag could be achieved through flying passenger airliners in extended formations. These savings were calculated to reach maximum values of 54 and 35% for subsonic and transonic speeds, respectively. However, of concern was the amount by which vortex tracking error may detract from these maximum values. The tracking error was theorised to dramatically reduce aerodynamic benefits, and its effect is amplified by atmospheric turbulence.

In addition, concerns of engine efficiency due to dynamic throttling, were raised. Dynamic throttling is necessary for an aircraft to maintain a beneficial position in the formation. Examples of dynamic throttling, due to station-keeping control in close proximity formations, were discussed. Results from NASA's AFF revealed that savings in induced drag did not directly correspond to fuel burn savings. For the test results presented, the

fuel burn savings were approximately 2% lower than induced drag savings. As a result, it was concluded that further investigations into the effect of tracking error on the benefits achieved by extended formations, are required. These investigations should take into account the effects of different levels of turbulence as well as engine performance under dynamic throttling.

Chapter 3

Formation Dynamics

3.1 Formation Aerodynamics

The concept of extended formations is still a relatively new idea. Therefore, there are only a few published works available [13, 24, 25], which explore the appropriate methods for modelling formation aerodynamics. Typically, these works have adopted, and modified, methodologies used to model close-proximity formations. Such approaches have included horseshoe vortex, vortex lattice and lifting line methods.

The first method presented in literature, Ning *et al.* [13], included a detailed wake model, which accounted for the effects of wake roll-up, viscous decay, vortex instabilities and disturbances due to turbulent gusts. This wake model, was then used to determine the induced velocities experienced by a trailing aircraft (and subsequent aircraft within the formation), thereby accounting for changes in the aerodynamic forces experienced by the trailing aircraft. Kless *et al.* [24], furthered this work through the inclusion of compressibility and roll trim effects.

Although these higher fidelity analyses have provided great insight into the aerodynamic interactions between aircraft, they are not necessarily appropriated for initial explorations into formation dynamics (as is the objective of this work), due to their complexity. Instead, a simpler approximate method is required; one that yields reasonable results and can be incorporated easily within a dynamic model. This is particularly true, considering the additional complexities introduced by the inclusion of engine transients, required to calculate fuel consumption, within the formation dynamics model (as is the requirement of this work).

Bizinos and Redelinghuys [25], recently developed a simplified model, which was used to analyse the effects of atmospheric turbulence on passenger comfort in extended formations. This model made use of horseshoe vortices to represent each aircraft of a twin-aircraft formation, and followed closely the methods of Blake and Multhopp [17], with some modifications.

The use of horseshoe vortex approach, greatly simplifies the aerodynamic modelling of the formation, whilst providing reasonable accuracy when compared to vortex lattice methods [17]. Therefore, this work has adopted the extended horseshoe vortex method developed by Bizinos and Redelnghuys, for inclusion in the dynamic formation model. The remainder of this section discusses this technique, with consideration to the context of this work.

3.1.1 Aircraft Representation

Figure 3.1 illustrates the horseshoe vortex representation of a twin-aircraft formation, of identical aircraft, as suggested by Bizinos and Redelinghuys [25]. As depicted, the lead aircraft is represented by a single horseshoe vortex only. It is unnecessary to include a horseshoe vortex to represent other aerodynamic surfaces (i.e. tailplane and tailfin) of the lead aircraft, because it is assumed that the streamwise separation between aircraft is sufficient such that the entire wake behind the lead aircraft has fully rolled up into two trailing vortices [13], and the lead aircraft's aerodynamics are not influenced by the trailing aircraft.

The trailing aircraft, on the other hand, makes use of a horseshoe vortex to represent each major aerodynamic surface; the main wing, tailplane and tailfin. This is necessary to determine the influence of the lead aircraft's trailing vortices, on the aerodynamic loads at each surface of the trailing aircraft; and thereby the effect on the trailing aircraft's motion in all 6 DOF.

However, for the purposes of this work, the tailfin vortex has been omitted (as in Figure 3.1), since it is assumed to only influence lateral aerodynamic loads (i.e. sideforce and yawing moment). This is because only the longitudinal formation dynamics will be considered in investigating fuel consumption in formation flight, in the context of this work.

Each horseshoe vortex includes a bound vortex, across the a span of each aerodynamic surface at its quarter chord, which is connected to two vortex filaments that propagate infinitely downstream. Assuming an elliptical lift distribution, the span of the bound vortex is $b_v = \frac{\pi}{4}b$. In addition, vortex decay is neglected, and therefore the trailing vortices are assumed to be constant in strength in the stream-wise direction.

3.1.2 Definition of the Vortex Filament

The bound and trailing vortices of the lead aircraft induce a flow field on the trailing aircraft, in accordance with the Biot-Savart law. Typically, the Biot-Savart law is defined in terms of the classical definition of the vortex profile, which is also commonly referred to as the Helmholtz profile. The Helmholtz profile, defines the tangential velocity of a point vortex as being directly and inversely proportional to its vortex strength and radius, respectively:

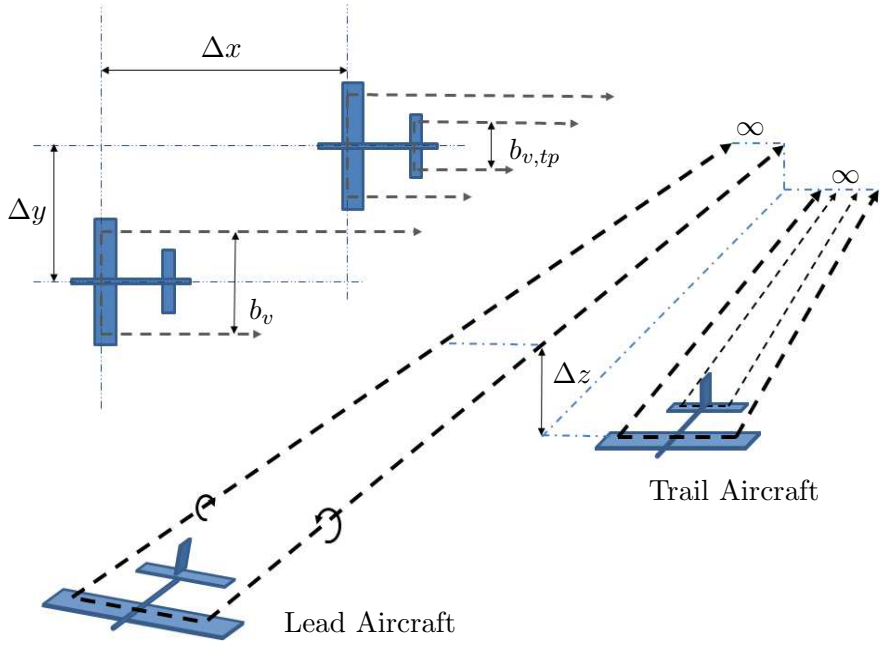


Figure 3.1: Schematic Diagram of the Extended Horseshoe Vortex Method

$$V_\theta\langle r\rangle = \frac{\Gamma}{2\pi r} \tag{3.1}$$

The Biot-Savart law, in terms of the Helmholtz profile, is given by Equation 3.2 with reference to Figure 3.2, [15]. This figure illustrates the change in velocity, due to the directed segment $\mathbf{dl}$, at point P, which is at distance r from the centre of a vortex filament of strength Γ.

$$\mathbf{dV} = \frac{\Gamma}{4\pi}\frac{\mathbf{dl}\times\boldsymbol{r}}{|\boldsymbol{r}|^3} \tag{3.2}$$

The use of the Helmholtz profile within the Biot-Savart law, as is clearly evident from Equation 3.2, results in singularities; for example, when the trailing vortex of the lead aircraft directly impinges on the bound vortex of the trailing aircraft, the induced velocity at that point tends to infinity. This is because the Helmholtz profile does not account for viscous flow near the centre of the vortex.

Instead, a better approximate is to assume that the vortex filament has a viscous core. Blake and Multhopp [17], and Bizinos and Redelinghuys [25] suggest replacing the Helmholtz profile with the Burnham Hallock profile [26], due to its ease of use (simple to integrate analytically) and reasonable agreement with experimental data. The Burnham Hallock profile is given by Equation 3.3, where r_c denotes the radius of the viscous core.

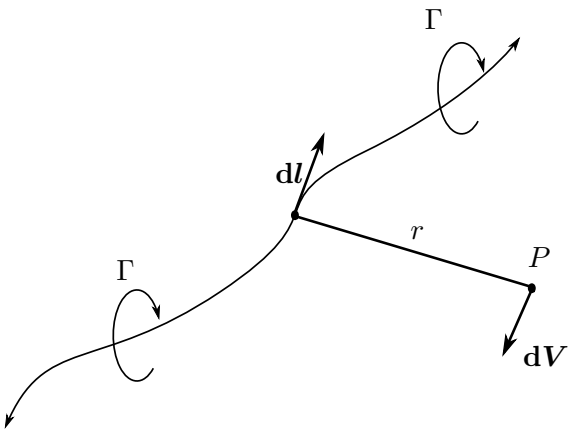


Figure 3.2: Flow induced according to the Biot-Savart law. Adapted from Anderson [15]

$$V_\theta\langle r\rangle = \frac{\Gamma}{2\pi}\frac{r}{(r^2 + r_c^2)} \tag{3.3}$$

Different core sizes may be used in defining this profile, and a core size of 3% of the wing's span (i.e. $r_c = 0.03b$) is used by both Blake and Multhopp, and Bizinos and Redelinghuys. This work aims to replicate the formation aerodynamics model of the latter, and therefore has adopted the same core size.

3.1.3 Induced Flow

The lead aircraft's bound vortex and trailing vortices, will all induce flow on the bound vortices (i.e. that on the main wing, tailplane and tailfin) of the trailing aircraft. However, Biznos [27] demonstrated that, for streamwise separations greater than 1 span, the flow induced (on the trailing aircraft's bound vortex) by the lead aircraft's bound vortex, is negligible in comparison to that induced by the lead aircraft's trailing vortices. This observation largely simplifies the calculation for determining the induced flow experienced by the trailing aircraft in extended formation flight. Therefore, only the flow induced by the lead aircraft's trailing vortices, on the trail aircraft's bound vortices, will be considered.

Consider the Biot-Savart law, with Helmholtz vortex profile, applied to a trailing vortex, as depicted in Figure 3.3. Figure 3.3, illustrates the change in velocity induced at a point P, by a directed line segment $\mathbf{dl}$ of a vortex filament of strength Γ. The vortex filament extends from point A to infinity, and the perpendicular distance between the vortex filament and point P is denoted h. The angle θ denotes angle between the vector radius $\boldsymbol{r}$ and the directed line segment of the vortex filament $\mathbf{dl}$.

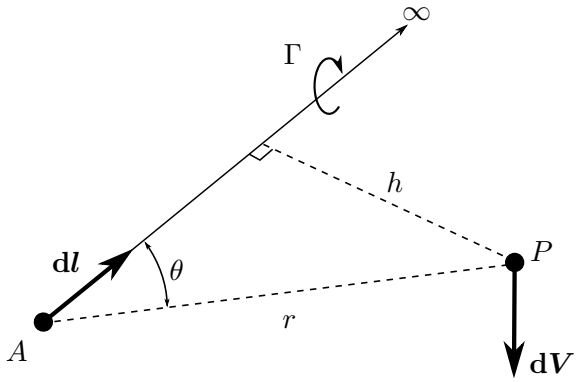


Figure 3.3: Biot-Savart law applied to a trailing vortex

The direction of the velocity induced at point P, is indicated to be downwards from the definition of the vector cross product and the right hand rule. Following the methodology of Anderson [15], the magnitude of this velocity may be found by integrating Equation 3.2:

$$V_{\theta_H} = \frac{\Gamma}{4\pi} \int_A^\infty \frac{\sin\theta}{r^2}\,\mathrm{d}l \tag{3.4}$$

The lengths of r, h and l may be related to one another by the angle θ:

$$r = \frac{h}{\sin\theta} \tag{3.5a}$$

$$l = \frac{h}{\tan\theta} \tag{3.5b}$$

$$\mathrm{d}l = -\frac{h}{\sin^2\theta}\mathrm{d}\theta \tag{3.5c}$$

Substituting the Equations 3.5a to 3.5c into Equation 3.4, yields the definite integral:

$$V_{\theta_H} = -\frac{\Gamma}{4\pi h} \int_{\theta_A}^{\pi} \sin\theta\,\mathrm{d}\theta \tag{3.6}$$

This may be easily integrated to give Equation 3.7, which may be used to determine the induced velocity by the trailing vortex filament on a point P on the bound vortex of the trailing aircraft.

$$V_{\theta_H} = \frac{\Gamma}{4\pi h}\left(1 + \cos\theta_A\right) \tag{3.7}$$

As mentioned, Equations 3.4 to 3.7 were derived in accordance with the Helmholtz vortex profile, and hence the subscript H attached to the symbol V_{θ_H}.

However, Equation 3.7 may be easily adjusted to accommodate the Burnham Hallock profile of Equation 3.3. The result is given by Equation 3.8, where the subscript BH denotes the use of the Burnham Hallock vortex profile.

$$V_{\theta_{BH}} = \frac{\Gamma}{4\pi}\left(\frac{h}{h^2 + r_c^2}\right)(1 + \cos\theta_A) \tag{3.8}$$

The flow induced on the bound vortices of a trailing aircraft is derived in Appendix A. The resulting flow is that which is induced by the lead aircraft's trailing vortices only, and does not include the flow induced by the lead aircraft's bound vortex. This is because Bizinos [27], demonstrated that its contribution was negligible, in comparison to that of the trailing vortices, for streamwise separations between aircraft greater than one span. Therefore the flow induced on the trailing aircraft is given by Equation A.12, which is repeated here for convenience:

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{jk,BH} = 2\left[\frac{\left(y + \Delta y + \frac{b_v}{2}\right)}{\left(y + \Delta y + \frac{b_v}{2}\right)^2 + \Delta z^2 + r_c^2} - \frac{\left(y + \Delta y - \frac{b_v}{2}\right)}{\left(y + \Delta y - \frac{b_v}{2}\right)^2 + \Delta z^2 + r_c^2}\right] \tag{3.9}$$

Equation 3.9 gives the downwash component of the induced flow; i.e. the vertical component in the inertial reference frame, assuming straight and level flight. Only this particular component is of interest, in the context of this work, as any lateral or longitudinal components are assumed not to have any effect the longitudinal motion of the aircraft. A particularly important result of Equation 3.9, is that the induced downwash is purely a function of the formation's geometry, thereby greatly simplifying its implementation. From here on, the downwash induced on the trailing aircraft k by the lead aircraft j using the Burnham Hallock profile, will simply be denoted as w_{jk}, thereby dropping the additional BH subscript.

To obtain the total induced flow experienced by the trailing aircraft, it is necessary to integrate Equation 3.9 along the span of the bound vortex. This integral has been termed the *influence factor*, and is defined as follows:

$$\sigma_{jk} = \left(\frac{4\pi}{\Gamma_j}\right) \int_{-\frac{b_v}{2}}^{+\frac{b_v}{2}} w_{jk}\, \mathrm{d}y \tag{3.10}$$

In keeping with the assumption of an elliptical lift distribution, Equation 3.10 requires numerical integration along the elliptical planform. However, in the aim of expressing the induced downash analytically, Bizinos and Redelinghuys [25] used the approximation of integrating Equation 3.10 along a rectangular planform of span b_v. The result of this integration is given by

Equation 3.11, with Δx, Δy, Δy and r_c dimensionalised according to the wingspan b:

$$\frac{\Delta x}{b} = \xi; \qquad \frac{\Delta y}{b} = \eta; \qquad \frac{\Delta z}{b} = \zeta; \qquad \frac{r_c}{b} = \mu$$

$$\sigma_{jk} = \left(\frac{4\pi}{\Gamma_j}\right) \int_{-\frac{b_v}{2}}^{+\frac{b_v}{2}} w_{jk}\,\mathrm{d}y = \ln \left| \frac{\left[\left(\eta - \frac{\pi}{4}\right)^2 + \zeta^2 + \mu^2\right]\left[\left(\eta + \frac{\pi}{4}\right)^2 + \zeta^2 + \mu^2\right]}{\left(\eta^2 + \zeta^2 + \mu^2\right)^2} \right| \tag{3.11}$$

Although this assumption may seem to be rather crude, Bizinos [27] was able to demonstrate a reasonable correlation between the integration over a rectangular planform versus and elliptical one. This correlation was particularly close for spanwises separations $\eta > 1$, which is likely to be the region of separation used by the formation. Therefore, the small loss in accuracy is justified by the convenience of the analytical solution, particularly in the context of this work.

3.1.4 Induced Aerodynamic Forces and Moments

Now that the induced flow has been determined, its effects on the aerodynamic forces and moments of the trailing aircraft may be examined. In this work, only the effects on lift drag and pitching moment will be discussed, as these are the loads that influence the decoupled longitudinal motion of the aircraft. Once again, this section will follow closely the derivations presented by Bizinos [27] and Bizinos and Redelinghuys [25], and the reader is referred to these works for detail about the formation influences on aircraft lateral motion.

Induced Lift

In quantifying the induced flow on the trail aircraft, it is first necessary to calculate the strength of the trailing vortices. Assuming that the lead aircraft is flying level, its lift must be equal to its weight. Knowing the lift, and assuming an elliptical lift distribution, the circulation may be obtained from the Kutta-Joukowski theorem:

$$\Gamma_j = \frac{L}{\rho_\infty V_\infty b_v} = \frac{2V_\infty b}{\pi AR} \tag{3.12}$$

Now, the change in lift of the trailing aircraft, is a result of a change in its effective angle of attack due to the induced downwash. At a point along the trailing aircraft's bound vortex, the induced downwash is obtained from Equation 3.9, and therefore the change in effective angle of attack is:

$$\delta\alpha_{k,eff} = -\tan\left(\frac{w_{jk}}{V_\infty}\right)$$

and assuming $V_\infty \gg w_{jk}$, the change in effective angle of attack is approximated as:

$$\delta\alpha_{k,eff} = -\left(\frac{w_{jk}}{V_\infty}\right)$$

Therefore, the change in lift coefficient at that point becomes:

$$\delta C_{l,k} = -C_{l_\alpha}\left(\frac{w_{jk}}{V_\infty}\right)$$

Applying this change across the span of the bound vortex:

$$\Delta C_{L,k} = -\frac{1}{b}\int_{-\frac{b_v}{2}}^{+\frac{b_v}{2}} C_{l_\alpha}\left(\frac{w_{jk}}{V_\infty}\right)\,\mathrm{d}y \tag{3.13}$$

The above equation may now be expressed in terms of the influence factor presented in Equation 3.11:

$$\Delta C_{L,k} = -\frac{C_{l_\alpha}\Gamma_j\sigma_{jk}}{4\pi V_\infty b} \tag{3.14}$$

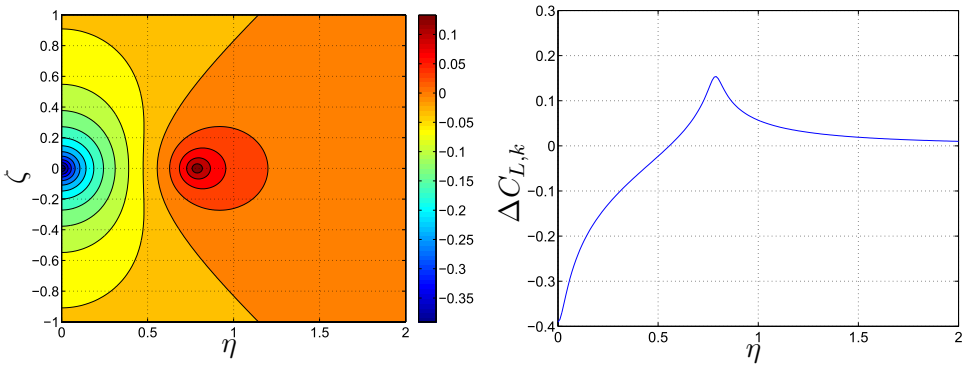
Finally, the change in trailing aircraft's the coefficient of lift, may be expressed in terms of the lead aircraft's coefficient of lift, as well as the geometric separation between aircraft. Substituting Equation 3.12 into Equation 3.14 yields:

$$\Delta C_{L,k} = -\frac{C_{l_\alpha}C_{L,j}\sigma_{jk}}{2\pi^2 AR} \tag{3.15}$$

Therefore, the change in lift may be calculated in terms of the lead aircraft's lift coefficient and the geometric separation between aircraft:

$$\Delta L_k = \Delta C_{L,k}\bar{q}_\infty S \tag{3.16}$$

Equation 3.15 was applied to a twin aircraft formation, of identical aircraft (see Appendix B for details on the aircraft), the results of which are presented in Figure 3.4. Figure 3.4a illustrates contours of constant $\Delta C_{L,k}$, for various vertical and lateral separations between aircraft. It is noted that the point $(\eta,\ \zeta) = (0,\ 0)$ corresponds to a geometry where the trailing aircraft is positioned directly behind the leader. At this point it is evident that the trailing aircraft is flying directly into the downwash of the leader, thereby experiencing the greatest reduction in lift. Moving outboard in the spanwise direction (i.e. into the upwash region of the leader), the trailing



(a) $\Delta C_{L,k}\,(\eta, \zeta)$ (b) $\Delta C_{L,k}\,(\eta, \zeta = 0)$

Figure 3.4: Change in lift coefficient, for a trailing aircraft, at various vertical and lateral separations

aircraft starts to experience improvements in lift. There exists a "sweet spot", at approximately $(\eta,\ \zeta) = (0.8,\ 0)$, where the greatest improvement in lift is achieved. At this position, the wing tips of the two aircraft are over-lapping slightly in the spanwise direction.

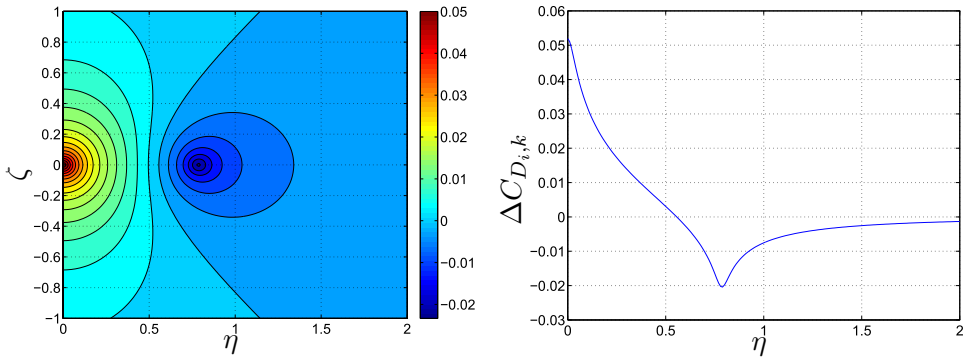
Figure 3.4b illustrates a cross-section of the contour plot at $\zeta = 0$, i.e. both aircraft are level with one another, which passes through the "sweet spot". This cross-section is useful, as it illustrates the gradients at which the lift coefficient varies for changes in lateral separation. These gradients have implications on how the aircraft will respond to perturbations, which becomes particularly important when trying to control the aircraft's position within the formation. Although desirable, it is not always practical to fly at the sweet spot. This is because, as is evident from Figure 3.4b, the gradients are at their steepest, and therefore it becomes very difficult to control the aircraft for perturbation about this position.

Induced Drag

Of particular importance in formation flight, is the change in induced drag achieved, as this allows the trailing aircraft to fly at a lower power setting. From Anderson [15] the Kutta-Joukowski theorem may be used to express the induced drag coefficient:

$$C_{D_i,k} = \frac{2}{V_\infty S} \int_{-\frac{b}{2}}^{+\frac{b}{2}} \Gamma_k(y)\, \alpha_i(y)\, \mathrm{d}y$$

Therefore, because the aircraft is modelled using a single horseshoe vortex to mimic an eliptical lift distribution, the above equation simplifies to:



(a) $\Delta C_{D_i,k}(\eta, \zeta)$ (b) $\Delta C_{D_i,k}(\eta, \zeta = 0)$

Figure 3.5: Change in induced drag coefficient, for a trailing aircraft, at various vertical and lateral separations

$$C_{D_i,k} = \frac{2\Gamma_k}{V_\infty S} \int_{-\frac{b_v}{2}}^{+\frac{b_v}{2}} \alpha_i(y)\, \mathrm{d}y$$

Therefore, the change in induced drag coefficient due to a change in induced angle of attack $\frac{w_{jk}}{V_\infty}$, is:

$$\Delta C_{D_i,k} = \frac{2\Gamma_k}{V_\infty^2 S} \int_{-\frac{b_v}{2}}^{+\frac{b_v}{2}} w_{jk}(y)\, \mathrm{d}y \tag{3.17}$$

Substituting in Equation 3.11, and representing the trailing aircraft's circulation in terms of its lift coefficient (as was done for the lead aircraft in Equation 3.12), Equation 3.17 becomes:

$$\Delta C_{D_i,k} = \frac{2\, C_{L,j}\, C_{L,k}\, \sigma_{jk}}{\pi^3\, AR} \tag{3.18}$$

Therefore, the change in induced drag may be calculated in terms of its, and the lead aircraft's, lift coefficient, as well as and geometric separation:

$$\Delta D_{i,k} = \Delta C_{D_{i,k}}\, \bar{q}_\infty\, S \tag{3.19}$$

Equation 3.18 was also applied to the same twin-aircraft formation as before. The results are depicted in Figure 3.5, following the same format as those presented for the lift. From this figure, it is evident that the trailing aircraft will experience the greatest increase in induced drag when positioned directly behind the leader. Moving outboard, there exists a sweet spot where the largest reduction in induced drag is achieved. As with the changes in induced lift, this sweet spot is located at a position where the wing-tips of the two aircraft are overlapping slightly. As was described in the case of induced lift, the gradients approaching the sweet spot are much steeper, and

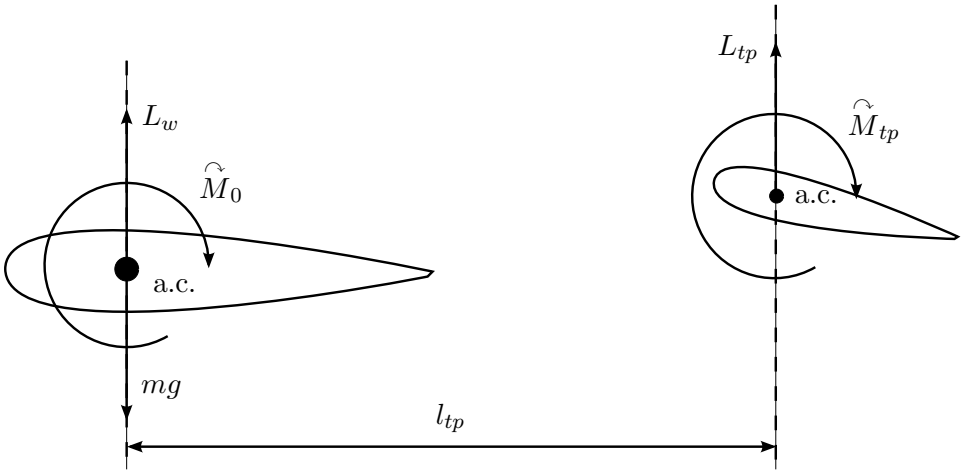


Figure 3.6: Schematic of pitching forces and moments

therefore it may be impractical for the trailing aircraft to fly in this position due to control concerns.

Induced Pitching Moment

In order to obtain an expression for the pitching moment in formation, it is first necessary to outline a general expression for the pitching moment in isolation. The development of such an expression is outlined by Cook [28], and forms the basis of the approach used in this work.

In developing a general pitching moment equation, it has been assumed that the aerodynamic contribution of the fuselage is negligible, and that the pitching moment is purely a result of the aerodynamic forces of the main wing and tailplane. Furthermore, steady and level flight is assumed. Therefore, thrust and drag forces are assumed to act through the c.g., thereby not influencing the pitching moment. In addition, this work assumes that the aerodynamic center of the main wing coincides with the center of gravity of the aircraft.

Figure 3.6, illustrates the the relevant forces and moments (as per the assumptions, drag forces have been omitted) acting on the main wing and tailplane. From this diagram, the pitching moment may be expressed as:

$$\hat{M}_{\text{c.g.}} = \hat{M}_0 + L_{tp} l_{tp} + \hat{M}_{tp} \tag{3.20}$$

Assuming the tailplane's airfoil section is symmetric, i.e. $\hat{M}_{tp} = 0$, the above equation may be expressed non-dimensionally:

$$C_m = C_{m_0} - C_{L,tp} \bar{V}_{tp} \tag{3.21}$$

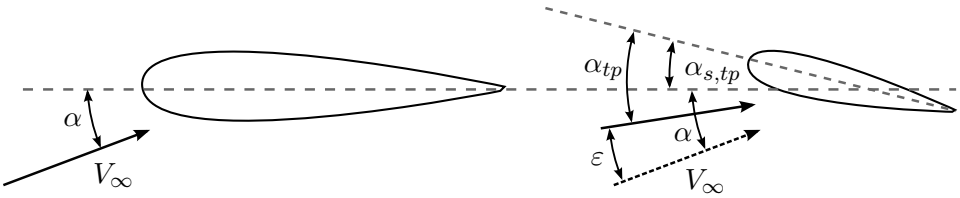


Figure 3.7: Tailplane aerodynamic angles for an aircraft in isolation. Adapted from c

Figure 3.7, illustrates the aerodynamic angles associated with the tailplane, ignoring elevator angles and trim tabs. From this figure, it is seen that the tailplane's angle of attack is:

$$\alpha_{tp} = \alpha + \alpha_{s,tp} - \varepsilon \tag{3.22}$$

where $\alpha_{s,tp}$ symbolises the tailplane's *setting angle*, and ε represents the local downwash seen by the tailplane. The tailplane's coefficient of lift is then expressed as:

$$C_{L,tp} = a_0 + a_1\alpha_{tp}$$

where a_0 and a_1 are aerodynamic coefficients, particular to the tailplane. However, the symmetrical airfoil assumption renders $a_0 = 0$, and the above equation becomes:

$$C_{L,tp} = a_1\alpha_{tp} \tag{3.23}$$

where a_1 is the tailplane's lift curve slope. Now, the local downwash at the tailplane, may be approximated as a function of the main wing's angle of attack and lift curve (assuming small perturbations):

$$\alpha - \varepsilon = \alpha\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) = \frac{C_{L,w}}{C_{l_\alpha}}\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) \tag{3.24}$$

Therefore, the tailplane's angle of attack is obtained by combining Equations 3.22 and 3.24

$$\alpha_{tp} = \frac{C_{L,w}}{C_{l_\alpha}}\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) + \alpha_{s,tp} \tag{3.25}$$

Finally, the induced downwash seen by the tailplane, due to the formation, may be included into Equation 3.25 (with reference to Figure 3.8):

$$\alpha_{tp,F} = \left(\frac{C_{L,kw}}{C_{l_\alpha}} - \delta\alpha_{i,tp}\right)\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) + \alpha_{s,tp} \tag{3.26}$$

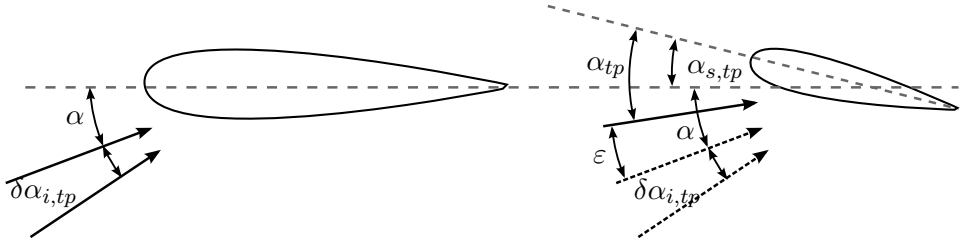


Figure 3.8: Tailplane aerodynamic angles for an aircraft in formation. Adapted from Bizinos and Redelinghuys [25]

The induced angle of attack $\delta\alpha_{i,tp}$, is the that which is induced on the main wing's bound vortex, but only over the portion that effects the tailplane. In other words, the portion of flow that passes over both the main wing and tailplane. This downwash is denoted $w_{jk,tp}$, and assuming $V_\infty \gg w_{jk,tp}$:

$$\delta\alpha_{i,tp} = \frac{w_{jk,tp}}{V_\infty}$$

and therefore:

$$\alpha_{tp,F} = \left(\frac{C_{L,kw}}{C_{l_\alpha}} - \frac{w_{jk,tp}}{V_\infty}\right)\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) + \alpha_{s,tp} \tag{3.27}$$

Combining Equations 3.21, 3.23 and 3.27, yields:

$$C_{m,k} = C_{m_0} - \bar{V}_{tp} a_1 \left[\left(\frac{C_{L,kw}}{C_{l_\alpha}} - \frac{w_{jk,tp}}{V_\infty}\right)\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) + \alpha_{s,tp}\right] \tag{3.28}$$

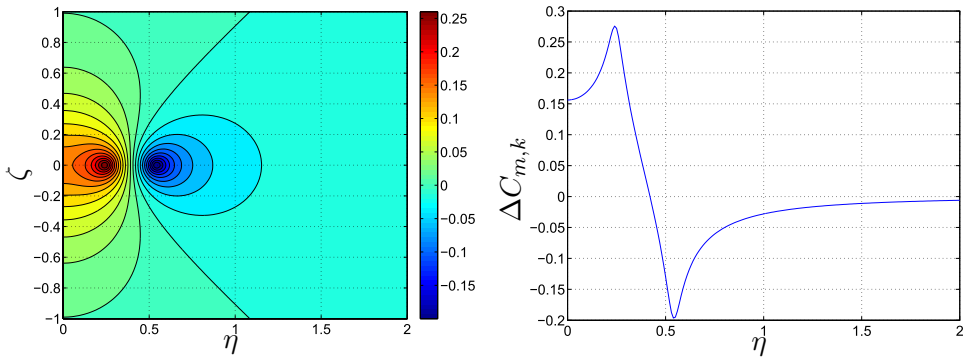
And comparing Equations 3.21 and 3.28, it is evident that the change in pitching moment coefficient is:

$$\Delta C_{m,k} = \bar{V}_{tp} a_1 \frac{w_{jk,tp}}{V_\infty}\left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) \tag{3.29}$$

The downwash $w_{jk,tp}$ is found by integrating along the trailing aircraft's bound vortex, but only for a portion equivalent to the span of the tailplane's bound vortex. Thus, the change in the tailplane's lift coefficient is given by Equation 3.30.

$$\Delta C_{L,tp} = -a_1 \frac{w_{jk,tp}}{V_\infty} = -\frac{a_1}{V_\infty b_{v,tp}} \int_{-\frac{b_{v,tp}}{2}}^{+\frac{b_{v,tp}}{2}} w_{jk}(y)\,\mathrm{d}y \tag{3.30}$$

And as before, an influence factor may be defined for the tailplane in particular:



(a) $\Delta C_{m,k}\,(\eta, \zeta)$ (b) $\Delta C_{m,k}\,(\eta, \zeta = 0)$

Figure 3.9: Change in pitching moment coefficient, for a trailing aircraft, at various vertical and lateral separations

$$\sigma_{jk,tp} = \left(\frac{4\pi}{\Gamma_j}\right) \int_{-\frac{b_{v,tp}}{2}}^{+\frac{b_{v,tp}}{2}} w_{jk}(y)\, \mathrm{d}y \tag{3.31}$$

Integrating over a rectangular planform results in:

$$\sigma_{jk,tp} = \ln \left| \frac{\left(\zeta^2 + \left(\eta - \frac{\pi}{8} - \frac{\pi}{8}\frac{b_{tp}}{b}\right)^2 + \mu^2\right)\left(\zeta^2 + \left(\eta + \frac{\pi}{8} + \frac{\pi}{8}\frac{b_{tp}}{b}\right)^2 + \mu^2\right)}{\left(\zeta^2 + \left(\eta - \frac{\pi}{8} + \frac{\pi}{8}\frac{b_{tp}}{b}\right)^2 + \mu^2\right)\left(\zeta^2 + \left(\eta + \frac{\pi}{8} - \frac{\pi}{8}\frac{b_{tp}}{b}\right)^2 + \mu^2\right)} \right| \tag{3.32}$$

Thus, the change in the tailplanes lift coefficient becomes:

$$\Delta C_{L,tp} = \frac{-2a_1\, C_{L,j}}{\pi^3\, AR\, \frac{b_{tp}}{b}}\, \sigma_{jk,tp} \tag{3.33}$$

which may be substituted back into Equation 3.29, to give a final expression for the trailing aircraft's change in pitching moment due to the presence of the lead aircraft's trailing vortices:

$$\Delta C_{m,k} = \frac{2\bar{V}_{tp}\, a_1\, C_{L,j}}{\pi^3\, AR\, \frac{b_{tp}}{b}} \left(1 - \frac{\mathrm{d}\varepsilon}{\mathrm{d}\alpha}\right) \sigma_{jk,tp} \tag{3.34}$$

Figure 3.9 depicts Equation 3.34, applied to the same twin-aircraft formation as in the case of the induced lift and drag effects. This figure depicts two peaks in the induced pitching moment. In-between these two peaks, the gradient of the changes in moment with lateral separation, Figure 3.9b, becomes very steep. Also, in this region the induced moment coefficient changes sign. Therefore, this region represents a lateral separation where control of the aircraft will become extremely difficult. It is interesting to

note that the pitching moment is, in a way, self-stabilizing. In regions of reduced lift (see Figure 3.4) the pitching coefficient increases (a positive value represents nose up), thereby increasing the aircraft's angle of attack. Therefore, if the correct amount of thrust is supplied, the aircraft will be able to easily counter the reduction in lift. The opposite is true for regions of increasing lift.

3.2 The Effects of Atmospheric Turbulence

At this point, atmospheric turbulence is introduced. It is necessary to model the turbulence gusts, to which the formation is subjected, in order to analyse the effects of formation flight in a more realistic environment. It is due to these turbulent gusts, that the aircraft in the formation will be perturbed from their respective stations. Such perturbations result in unique fluctuations in the aerodynamic forces and moments of each aircraft, thereby inducing dynamics within the formation. It is because of theses dynamics that the aircraft's engines must be dynamically throttled, in an attempt to maintain its position within the formation and, therefore, the aerodynamic benefit received.

3.2.1 Atmospheric Turbulence Model

This work makes use of the atmospheric turbulence model found in Simulink's Aerospace Blockset. This model has been detailed by Gage [29], which is briefly summarised here.

Atmospheric turbulence is considered to be a stationary, homogeneous and, therefore, ergodic stochastic process. Because it is assumed stationary, the gusts experienced by the aircraft are a result of the aircraft's motion relative to the atmosphere.

The turbulent atmospheric gusts may be generated by applying appropriate filters to a unit variance, band-limited white noise signal. Such filters are derived using the Dryden or Von Kármán spectra, in accordance with either standard MIL-F-8785C [30] or MIL-HDBK-1797 [31]. The Simulink turbulence model, allows for any combination of spectra and standards, as the spectra definitions differ slightly between standards.

The Military Specification MIL-HDBK-1797 has been selected for this particular work, as it is a significantly more recent specification than the Military Specification MIL-F-8785C. In addition, the Von Kármán spectra has been selected, as its results have been shown to correlate much more closely to experimental data, than those of the Dryden spectra [32].

The Von Kármán spectra from MIL-HDBK-1797, as presented by Gage [29], are given in Equations 3.35 to 3.40. In these equations the spectra is given as a function temporal frequency ω with constants representing the root-mean-square intensity σ, and scale length L. Equations 3.35 and 3.36,

represent the longitudinal spectra of the gust u_g and angular rate about the x-axis, respectively.

$$\Phi_u(\omega) = \frac{2\sigma_u^2 L_u}{\pi V} \frac{1}{\left[1 + \left(1.339 L_u \frac{\omega}{V}\right)^2\right]^{5/6}} \tag{3.35}$$

$$\Phi_p(\omega) = \frac{\sigma_w^2}{2VL_w} \frac{0.8\left(\frac{2\pi L_w}{4b}\right)^{1/3}}{1 + \left(\frac{4b\omega}{\pi V}\right)^2} \tag{3.36}$$

Equations 3.37 and 3.38, represent the longitudinal spectra of the gust v_g and angular rate about the y-axis, respectively.

$$\Phi_v(\omega) = \frac{2\sigma_v^2 L_v}{\pi V} \frac{1 + \frac{8}{3}\left(2.678 L_v \frac{\omega}{V}\right)^2}{\left[1 + \left(2.678 L_v \frac{\omega}{V}\right)^2\right]^{11/6}} \tag{3.37}$$

$$\Phi_r(\omega) = \frac{\mp\left(\frac{\omega}{V}\right)^2}{1 + \left(\frac{3b\omega}{\pi V}\right)^2} \Phi_v(\omega) \tag{3.38}$$

Equations 3.39 and 3.40, represent the longitudinal spectra of the gust w_g and angular rate about the z-axis, respectively.

$$\Phi_w(\omega) = \frac{2\sigma_w^2 L_w}{\pi V} \frac{1 + \frac{8}{3}\left(2.678 L_w \frac{\omega}{V}\right)^2}{\left[1 + \left(2.678 L_w \frac{\omega}{V}\right)^2\right]^{11/6}} \tag{3.39}$$

$$\Phi_q(\omega) = \frac{\mp\left(\frac{\omega}{V}\right)^2}{1 + \left(\frac{4b\omega}{\pi V}\right)^2} \Phi_w(\omega) \tag{3.40}$$

For applications at medium to high altitudes, the intensity is depicted by Figure 3.10, where $\sigma_u = \sigma_v = \sigma_w$. The intensities for light moderate and severe turbulence, which will be considered in this work, have been indicated on Figure 3.10. The scale lengths (according to the Von Kármaán spectra and MIL-HDBK-1797) are:

$$L_u = 2L_v = 2L_w = 2500ft = 762m$$

From the above theory, the Simulink turbulence block outputs the turbulence gusts u_g, v_g and w_g, as well as angular rates p_g, r_g and q_g, at each time step. It is these parameters that perturb the aircraft and the formation as a whole. The following sections, will discuss such perturbations according these parameters.

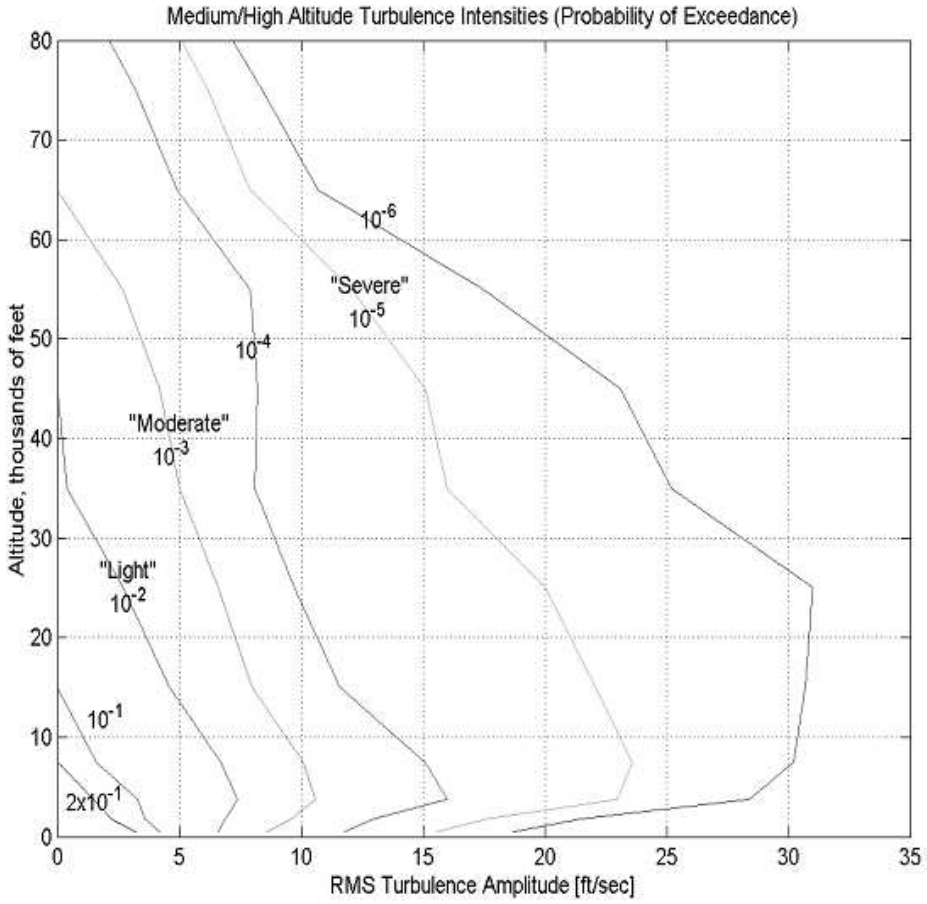


Figure 3.10: Turbulence RMS intensities for turbulence at medium to high altitudes. Reprinted from Gage [29]

3.2.2 Effects on Isolated Aircraft

First, the case of the isolated aircraft is considered. Very simply, the isolated aircraft is approximated as a single point, thus neglecting any gust gradients across the airframe. This approximation assumes that the airframe is relatively small in comparison to the wavelength of the atmospheric turbulence. Therefore, the influence of the turbulence may be seen as a change in free stream conditions, from the aircraft's perspective.

In addition, the aircraft's response due to the angular rates is relatively small in comparison to its response to the turbulent gusts. Therefore, the effects of the turbulent angular rates are assumed to have a negligible effect on the aircraft, and are omitted from this work.

Finally, only the longitudinal aircraft dynamics are considered, and therefore only the vertical w_g and longitudinal u_g gust components effect the aircraft. However, $u_g \ll V_\infty$, and therefore its effects are assumed to be negligible.

Therefore, according to the above assumptions, the only parameter of concern is the vertical gust component w_g. The vertical gust component alters the aircraft's effective angle-of-attack by :

$$\Delta\alpha_g = \tan^{-1}\left(\frac{w_g}{V_\infty}\right)$$

Which, assuming $w_g \ll V_\infty$, simply becomes:

$$\Delta\alpha_g = \frac{w_g}{V_\infty} \tag{3.41}$$

This change in angle-of-attack, directly alters the aircraft's aerodynamic forces and moments according to the aerodynamic coefficients C_{L_α}, C_{D_α} and C_{m_α}.

3.2.3 Effects on Formation Induced Forces and Moments

The lead aircraft's trailing vortices will be shifted by the turbulence. This shift, results in a change in relative positioning between the trail aircraft and the trailing vortices of the leader. As was discussed earlier in this chapter, the induced aerodynamic forces and moments on the trail aircraft, are a function of its position relative to the leader's trailing vortices. Therefore, fluctuations in this relative positioning will result in fluctuations in the induced forces and moments experienced by the trailing aircraft. Therefore, a method of calculating the change in this relative positioning, due to atmospheric turbulence, is required.

This work will follow the method presented by Bizinos and Redelinghuys [25], for determining the effective separation between the leaders trailing vortices and the horseshoe vortex of the trail aircraft.

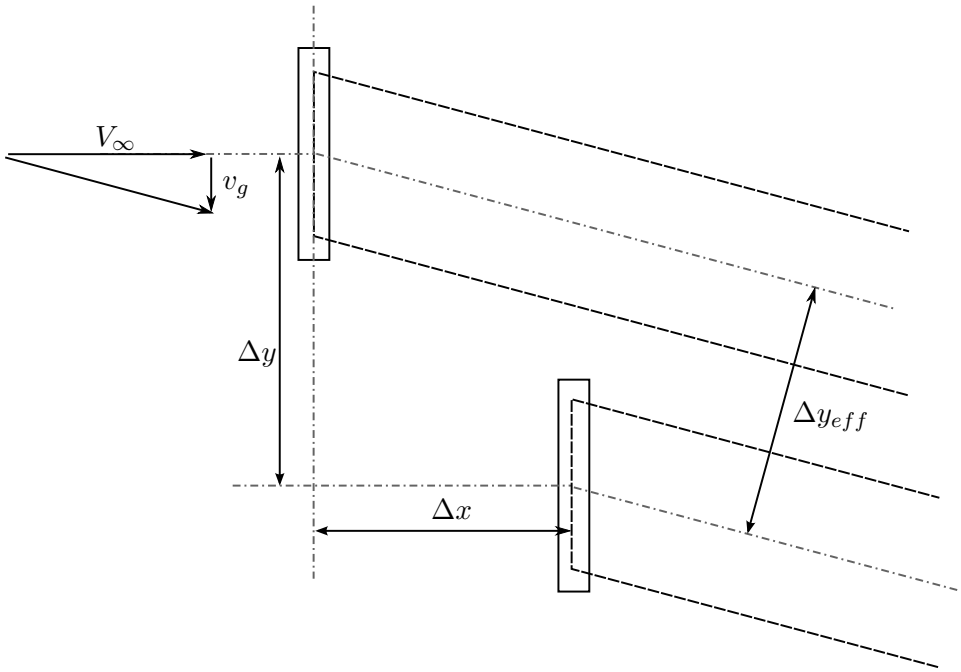


Figure 3.11: Effective lateral separation. Adapted from Bizinos and Redelinghuys [25]

Firstly, despite large streamwise separations between aircraft, it is assumed that the formation, as a whole, is much smaller than the wavelength of the turbulence. Hence, the trailing aircraft will experience the exact same turbulent gusts as the leader, at any instance in time. Secondly, it is assumed that the trailing vortices shift perfectly with the atmosphere. Therefore, for any instance in time, all trailing vortices of the formation lie parallel with the turbulent free-stream velocity. Thirdly, it is assumed that $u_g \ll V_\infty$, and hence its effects are negligible. Finally, it is assumed that the formation's response to turbulent angular rates is negligible in comparison to its response to turbulent gusts.

Given these assumptions, the effective separation between the trail aircraft and the leader's trailing vortices, is illustrated by Figures 3.11 and 3.12, for lateral and vertical gusts respectively.

Therefore, the effective lateral separation, with reference to Figure 3.11, is:

$$\Delta y_{eff} = \sqrt{\Delta x^2 + \Delta y^2}\, \sin\left[\tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) - \tan^{-1}\left(\frac{v_g}{V_\infty}\right)\right] \tag{3.42}$$

and similarly, the effective vertical separation, with reference to Figure 3.12, is:

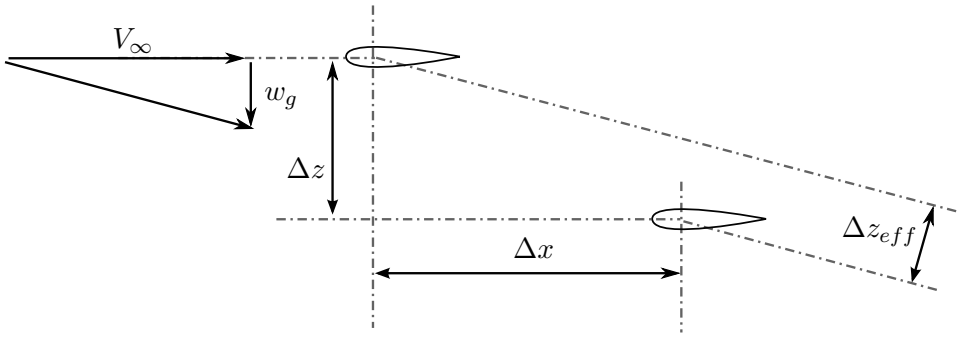


Figure 3.12: Effective vertical separation. Adapted from Bizinos and Redelinghuys [25]

$$\Delta z_{eff} = \sqrt{\Delta x^2 + \Delta z^2}\ \sin\left[\tan^{-1}\left(\frac{\Delta z}{\Delta x}\right) - \tan^{-1}\left(\frac{w_g}{V_\infty}\right)\right] \tag{3.43}$$

Dimensionalising Equations 3.42 and 3.43 according to wingspan b, results in Equations 3.44 and 3.45, respectively.

$$\eta_{eff} = \sqrt{\xi^2 + \eta^2}\ \sin\left[\tan^{-1}\left(\frac{\eta}{\xi}\right) - \tan^{-1}\left(\frac{v_g}{V_\infty}\right)\right] \approx \eta - \xi\left(\frac{v_g}{V_\infty}\right) \tag{3.44}$$

$$\zeta_{eff} = \sqrt{\xi^2 + \zeta^2}\ \sin\left[\tan^{-1}\left(\frac{\zeta}{\xi}\right) - \tan^{-1}\left(\frac{v_g}{V_\infty}\right)\right] \approx \zeta - \xi\left(\frac{v_g}{V_\infty}\right) \tag{3.45}$$

Thus, the induced aerodynamic forces and moments on the trail aircraft, in turbulence, may be obtained by using the effective separations of Equations 3.44 and 3.45, in Equations 3.15, 3.18 and 3.34.

3.3 Individual Aircraft Models

Each aircraft within the formation is subject to individual dynamics. This section discusses the models used to simulate the lead and trail aircraft individually.

3.3.1 3DOF Equations of Motion

Each aircraft within the formation, is modelled in terms of its decoupled longitudinal equations of motion, using the 3DoF model contained within Simulink's Aerospace Blockset. The equations of motion, in body axis, are given in Equations 3.46 to 3.49.

$$\dot{u}_B = \frac{F_{x,B}}{m} - q_B w_B - g \sin\theta_B \tag{3.46}$$

$$\dot{w}_B = \frac{F_{z,B}}{m} - q_B u_B - g \cos\theta_B \tag{3.47}$$

$$\dot{q}_B = \frac{\overset{\curvearrowright}{M}_B}{I_y} \tag{3.48}$$

$$\dot{\theta}_B = q_B \tag{3.49}$$

The above equations have not been linearised, and are solved numerically in the Simulink environment [1]. Upon solving the above equations, the aircraft's change in position and orientation may be calculated.

It now remains for the forces $F_{x,B}$ and $F_{z,B}$, to be derived for an aircraft in isolation (i.e. the lead aircraft) and formation (i.e. the trail aircraft). Assuming small perturbations about a cruise trim condition (see Appendix B), the aerodynamic forces and moments for the lead and trail aircraft are given by Equations 3.50 to 3.52 and 3.53 to 3.55, respectively.

$$L_j = \bar{q}S\left[C_{L_0} + C_{L_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{L_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{L_M}\Delta M + C_{L_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{L_q}\Delta q\right] \tag{3.50}$$

$$D_j = \bar{q}S\left[C_{D_0} + C_{D_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{D_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{D_M}\Delta M + C_{D_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{D_q}\Delta q\right] \tag{3.51}$$

$$\overset{\curvearrowright}{M}_j = \bar{q}S\bar{c}\left[C_{m_0} + C_{m_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{m_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{m_M}\Delta M + C_{m_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{m_q}\Delta q\right] \tag{3.52}$$

$$L_k = \bar{q}S\left[C_{L_0} + C_{L_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{L_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{L_M}\Delta M + C_{L_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{L_q}\Delta q + \Delta C_{L,Fk}\right] \tag{3.53}$$

$$D_k = \bar{q}S\left[C_{D_0} + C_{D_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{D_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{D_M}\Delta M + C_{D_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{D_q}\Delta q + \Delta C_{D_i,Fk}\right] \tag{3.54}$$

[1] These equations are well documented in most texts on flight mechanics, and therefore the reader is referred to Cook [28] for a more detailed description.

$$\overset{\curvearrowright}{M}_k = \bar{q}S\bar{c}\left[C_{m_0} + C_{m_\alpha}\Delta\alpha + \frac{\bar{c}}{2u}C_{m_{\dot{\alpha}}}\Delta\dot{\alpha} + C_{m_M}\Delta M + C_{m_{\delta e}}\Delta\delta e + \frac{\bar{c}}{2u}C_{m_q}\Delta q + \Delta C_{m,Fk}\right] \tag{3.55}$$

In Equations 3.53 to 3.55, the induced aerodynamic forces and moments, $\Delta C_{L,Fk}$, $\Delta C_{D_i,Fk}$ and $\Delta C_{m,Fk}$, are given by Equations 3.15, 3.18 and 3.34, respectively.

It is assumed that both aircraft are in level and straight flight, and therefore the forces and moments of Equations 3.50 to 3.55 correlate to the forces in the inertial reference frame. Therefore, the aerodynamic forces and moments of Equations 3.50 to 3.55, need transferring from inertial to body axis. This may be done using the general formula below:

$$\boldsymbol{F}_B = \begin{bmatrix} L & D \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3.56}$$

Therefore, using the above expression, Equations 3.50 to 3.55 may be inserted into the equations of motion; Equations 3.46 and 3.47. It is noted that the pitching moment $\overset{\curvearrowright}{M}$ is the same in body and inertial axis, assuming straight and level flight, and may therefore be applied in Equation 3.48.

3.3.2 Aircraft Trim Conditions

It is necessary to briefly discuss the trim conditions of the aircraft within the formation, as this defines the formation's operation in still atmospheric conditions. The trim condition of the lead aircraft is the same as that of an aircraft in isolation. However, the trail aircraft must establish a new trim condition to account for the induced aerodynamic forces and moments it experiences in the presence of the leader's wake.

In order for the trail aircraft to maintain level flight and match the leader's airspeed, its incidence angle, elevator deflection and throttle setting must be altered.

The new values for these parameters are determined by resolving the forces (in the vertical and horizontal directions) and moments, depicted in Figure 3.13, such that they equate to zero. Considering steady, straight and level flight, the aerodynamic forces and moments of Equations 3.53 to 3.55 are simplified to Equations 3.57 to 3.59.

$$L_{k,\text{trim}} = \bar{q}S\left[C_{L_0} + C_{L_\alpha}\Delta\theta_{\text{trim}} + C_{L_{\delta e}}\Delta\delta e_{\text{trim}} + \Delta C_{L,Fk}\right] \tag{3.57}$$

$$D_{k,\text{trim}} = \bar{q}S\left[C_{D_0} + C_{D_\alpha}\Delta\theta_{\text{trim}} + C_{D_{\delta e}}\Delta\delta e_{\text{trim}} + \Delta C_{D_i,Fk}\right] \tag{3.58}$$

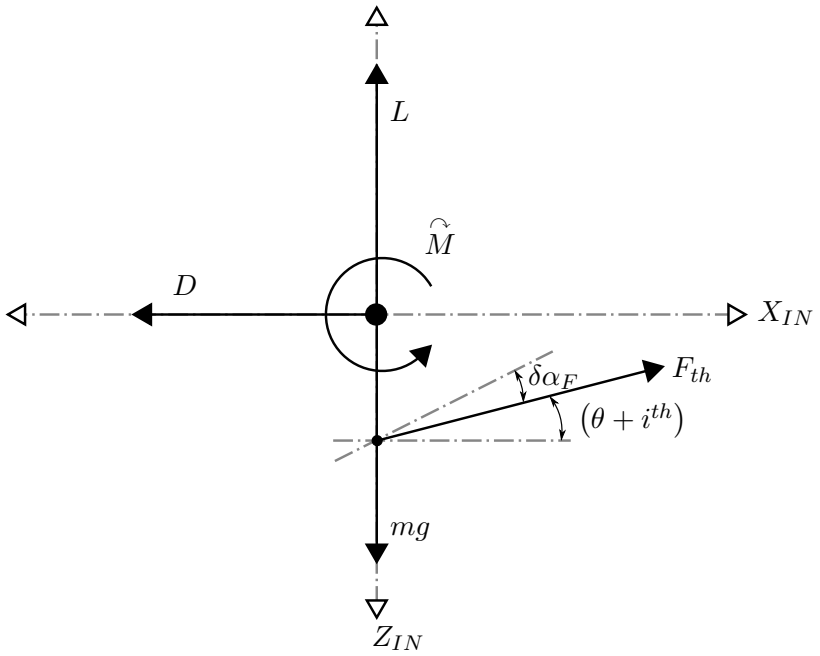


Figure 3.13: Trimmed condition force diagram

$$\overset{\curvearrowright}{M}_{k,\text{trim}} = \bar{q}S\bar{c}\left[C_{m_0} + C_{m_\alpha}\Delta\theta_{\text{trim}} + C_{m_{\delta e}}\Delta\delta e_{\text{trim}} + \Delta C_{m,Fk}\right] \tag{3.59}$$

To determine the trimmed incidence ($\theta_{k,trim} = \theta_j + \Delta\theta_{\text{trim}}$), elevator deflection ($\delta e_{k,trim} = \delta e_j + \Delta\delta e_{\text{trim}}$) and thrust $F_{th,\text{trim}}$, the vertical forces horizontal forces and moment must be resolved according to Equations 3.60 to 3.62.

$$L_{k,\text{trim}} + F_{th,\text{trim}} \sin\left(\theta_{k,trim} + i^{th}\right) - mg = 0 \tag{3.60}$$

$$D_{k,\text{trim}} - F_{th,\text{trim}} \cos\left(\theta_{k,trim} + i^{th}\right) = 0 \tag{3.61}$$

$$\overset{\curvearrowright}{M}_{k,\text{trim}} - F_{th,\text{trim}}\, l_{th} = 0 \tag{3.62}$$

An analytical solution to the above equations may be achieved more easily by assuming $\Delta\theta_{\text{trim}}$ to be small, thereby linearising any trigonometric terms; i.e. $\cos(\Delta\theta_{\text{trim}}) \approx 1$ and $\sin(\Delta\theta_{\text{trim}}) \approx \Delta\theta_{\text{trim}}$. This approximation has been used in this work to determine the trim conditions of the trailing aircraft.

3.4 2DOF Formation Dynamics Model

The size of the formation modelled in this work, has been limited to two identical aircraft (see Appendix B for aircraft details). Additional aircraft have not been included, because of the complexities introduced, not only in the formation dynamics, but in particular the aerodynamic interactions between aircraft. A more sophisticated model than the extended horseshoe vortex method, presented earlier, is required to account for vortex interactions that occur along a formation with multiple aircraft.

To model aircraft and a formation dynamically in all 6 DOF, requires highly sophisticated control system. The development of such a control system/autopilot falls well outside the scope of this work. Therefore, some crude simplifications are made, with the aim of producing a model that allows for initial explorations into formation dynamics.

As has already been discussed, the individual aircraft are modelled in terms of their decoupled longitudinal equations of motion. Therefore, its is assumed that the lateral motion of each aircraft is controlled perfectly. This is a very crude assumption, particularly considering the trail aircraft, because of the asymmetry in aerodynamic forces and moments in the presence of another aircraft's wake. A subsequent result of this assumption, is that the spanwise separation η between the aircraft is perfectly maintained.

Therefore, the formation dynamics may be modelled in terms of 3DOF aircraft models, which are given the freedom to accelerate in the longitudinal and vertical directions, as well as pitching about the aircraft's c.g..

However, to control such a formation still requires a fairly sophisticated control system. Such a control system will have to simultaneously manipulate engine throttle and elevator settings, in order to maintain the desired vertical and streamwise separation between aircraft. The development of this control system, despite its limitation to longitudinal motion, is still considered to be too complex for inclusion within the scope of this work.

Therefore, another simplification may be made, by assuming that both aircraft's vertical motion is somehow controlled perfectly. This assumption constrains both aircraft to fly at a fixed altitude and vertical separation. This final assumption restricts each aircraft to two degrees-of-freedom, where the aircraft is allowed to pitch and accelerate in the streamwise direction.

The resulting 2DOF formation model, despite such crude assumptions, has some merit in that it is analogous to wind tunnel experimentation. This "wind tunnel approach" to modelling formation dynamics, is depicted in Figure 3.14. As illustrated in this figure, the lead and trail aircraft are each allowed to pitch about their c.g., as well as accelerate in along the longitudinal inertial axis, only. It is noted, however, that the aircraft and formation is subject to three dimensional turbulence. Therefore, the induced aerodynamic forces and moments on the trail aircraft are still determined according to the effective separation parameters, as was discussed in Section

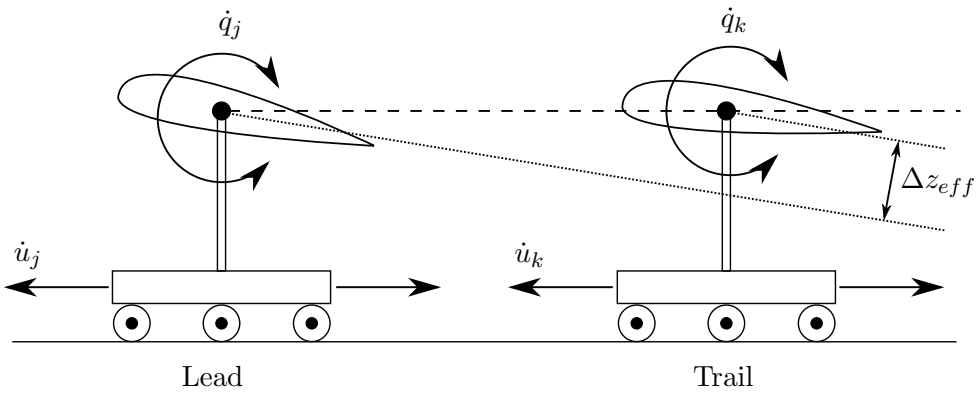


Figure 3.14: 2DOF Wind tunnel approach to formation dynamic modelling

3.2.3. In addition, the individual aircraft are still subject to direct influence by the atmospheric turbulence vertical gusts, as was discussed in Section 3.2.2.

Figure 3.14, also depicts the aircraft as flying level with one another, but this is not a requirement of the simplified wind tunnel approach, as aircraft may be fixed at different altitudes. However, this scenario has been used in this work, because when the the aircraft are level with one another (i.e. $\zeta = 0$) the greatest benefit may be realised. This is evident from Figures 3.4 and 3.5.

3.5 Controlling Streamwise Separation

It is now necessary to control the streamwise separation between aircraft, in the presence of atmospheric turbulence, in order to maintain the formation's geometry and aerodynamic benefit.

It will be assumed that the lead aircraft flies with a constant throttle setting, thereby avoiding any dynamic throttling and operating at maximum efficiency. Therefore the atmospheric turbulence will dictate the longitudinal acceleration of the aircraft. In addition, the atmospheric turbulence also effects the pitch rate of the lead aircraft. These dynamics in turn influence the lift force of the leader, which dictates the strength of its trailing vortices, in accordance with the Kutta-Joukowski theorem. The fluctuations in vortex strength will in turn influence the aerodynamic forces and moments induced on the trail aircraft.

The trail aircraft, under the influences of the leader's wake, must manipulate its throttle setting to maintain its desired position relative to the leader. A simplified schematic of this control system is depicted in Figure 3.15. This figure illustrates a negative feedback control loop, which attempts to maintain a desired streamwise separation between the aircraft.

The fuel controller of the system in Figure 3.15, decides the throttle

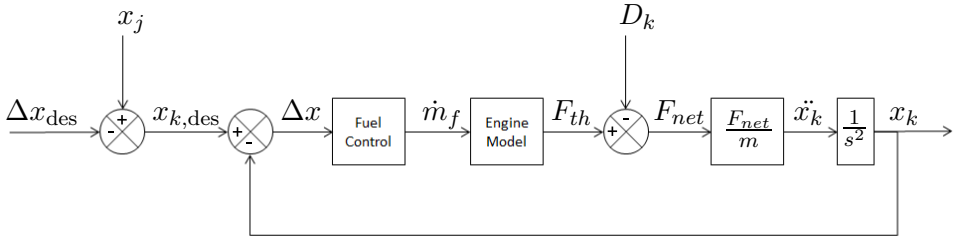


Figure 3.15: Simplified streamwise separation control loop

setting according to the error in separation between aircraft. A simple proportional gain controller is used, and is defined as:

$$\dot{m}_f = G_f \left(x_{k,\text{des}} - x_k \right) + \dot{m}_{f,\text{trim}} \tag{3.63}$$

In Equation 3.63, $\dot{m}_{f,\text{trim}}$ is the fuel flow rate setting at the trailing aircraft's trim conditions, when flying at the desired separation from the leader. G_f is a constant, which represents the proportional gain used in altering the fuel flow rate to the engine. The controller has not been given any rate or saturation limits, as these are governed by the engine's controller, which is discussed in Section 5.5.

3.6 Summary

This chapter introduced a method for modelling the aerodynamic benefits received by a trailing aircraft in formation flight. The model, referred to as the extended horseshoe vortex method, represents the leader as a single horseshoe vortex, whilst the trail aircraft is represented by a separate horseshoe vortex on its main wing and tailplane. The forces and moments induced on the trail aircraft were shown to be a function of the leader's vortex strength and geometrical separation between aircraft.

Next, atmospheric turbulence was introduced along with Simulink's turbulence block. The Von Kármán spectra was selected to model turbulence using the Simulink turbulence block. The effects of the turbulent gusts on an individual aircraft's longitudinal forces and moments was then explored. It was assumed that only the vertical turbulent gusts would effect an isolated aircraft, by altering its effective angle of attack. Following that discussion, the effects of turbulent gusts on the formation induced forces and moments were considered. Here, it was assumed that the trailing vortices would move perfectly and instantaneously with the turbulent atmosphere, and therefore would alter the effective separation between the leader's trailing vortex and the trailing aircraft. A change in the effective separation in turn alters the induced forces and moments.

The individual aircraft dynamic models were then discussed. These models are based on the decoupled longitudinal equations of motion. Aircraft trim conditions, particularly for the trailing aircraft, were considered. The formation dynamic model was then explained, indicating a 2 DOF model, referred to as the wind tunnel approach. Here, the aircraft are assumed to fly at a fixed vertical and lateral separation. However, the aircraft do have freedom to pitch and accelerate in the longitudinal direction.

Finally, the a control system to manage the streamwise separation between aircraft was considered. The lead aircraft is assumed to fly with a fixed throttle setting, and it is left to the trail aircraft to maintain its desired position relative to the leader. A proportional gain controller was used to manage the trail aircraft's throttle setting. The controller managers this throttle setting according to the error in separation between the aircraft.

Next, it is necessary to develop an engine model that is capable of predicting engine performance due to the dynamic throttling required in formation flight.

Chapter 4

Gas Turbine Engines

This chapter serves to introduce the gas turbine engines used in aircraft propulsion. The first section briefly describes the operation of gas turbine engines, with focus on thermodynamic cycles and performance characteristics. The second section presents the various simulation methods, that have existed since the engine's conception, for predicting the behaviour of engine dynamics. The section on simulation techniques is rather detailed in comparison to the others, as the purpose of this work was to develop a transient engine model for predicting fuel consumption due to dynamic throttling. Finally, the last section provides a summary of the chapter, which serves to highlight the significant details presented in this literature review.

4.1 Basic Operating Fundamentals

It is first important to have a look at the general layout of aircraft engines and their main components, as this defines their operation. Once the layout is defined, the engine's operation may be understood and its subsequent thermodynamic cycle and performance characteristics examined. This section aims to provide a brief, and simplified, overview of the gas turbine operation, necessary for understanding the following literature review on simulation of engine transients. The reader is referred to more comprehensive texts [33,34] for an in-depth description of gas turbine operation.

4.1.1 Engine Configurations and Layouts

A turbojet engine is the simplest form of a gas turbine engine used in propulsion. It is comprised of an intake, compressor, combustion chamber, turbine and exhaust nozzle. The compressor and turbine are coupled mechanically by a common shaft, often referred to as a spool. This simple form makes up a single-spool turbojet engine, and is illustrated in Fig. 4.1.

To improve performance, an additional compressor and turbine may be

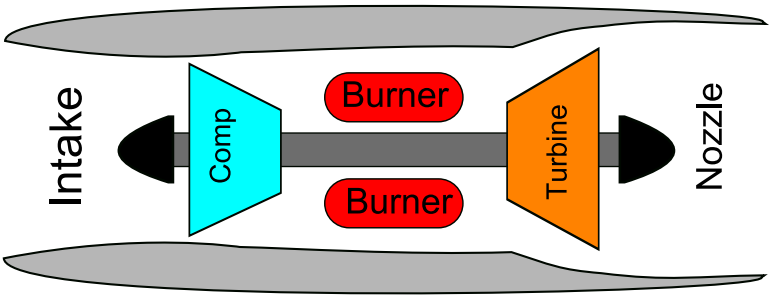


Figure 4.1: Typical single-spool turbojet engine configuration

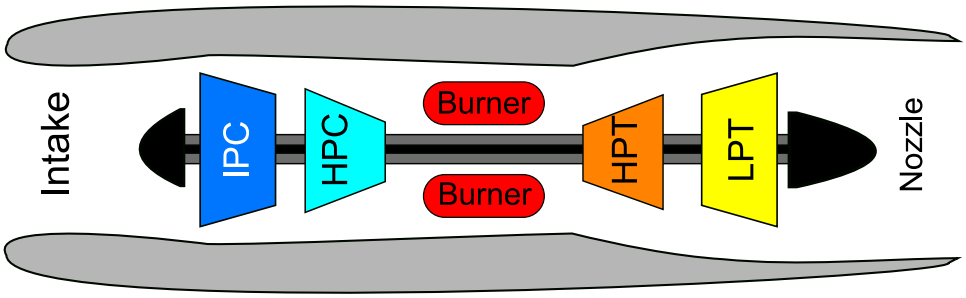


Figure 4.2: Typical twin-spool turbojet engine configuration

added to a turbojet engine. The additional components are coupled to one another on a separate spool which runs concentrically to the first. This allows for compressor-turbine pairs to run at different speeds to accommodate different pressures, thereby increasing the overall pressure ratio of the engine. The components adjacent to the combustor, on either side, operate at high pressures. Therefore, the spool, to which they are coupled, is referred to as the high-pressure spool. Conversely, there exist a low pressure spool, to which the outermost components are coupled. More than two-spools may be incorporated to a turbojet configuration, but it is only common place in turbofan engines. In the instance of a three-spool engine, the middle spool is referred to as the intermediate pressure spool. A twin-spool turbojet engine is illustrated in Fig. 4.2.

To improve efficiency, a fan may be incorporated at the front of a turbojet configuration. In so doing, the stream exiting the fan is split in two. A portion of the flow passes through the "core" of the engine, flowing through subsequent compressors, combustion chambers and turbines. The other portion flows axially around the core of the engine and is referred to as "bypass flow". Hence the term "bypass engine". Bypass engines require more than one-spool, due to the slower operating speed of the fan in comparison to the other compressors. Turbofan engines are generally twin or three-spool engines.

The bypass flow may be exhausted via its own nozzle (secondary nozzle),

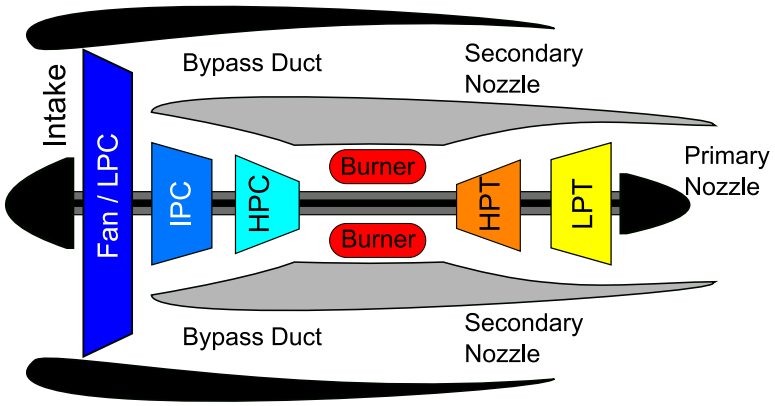


Figure 4.3: Typical configuration of a twin-spool high-bypass turbofan engine

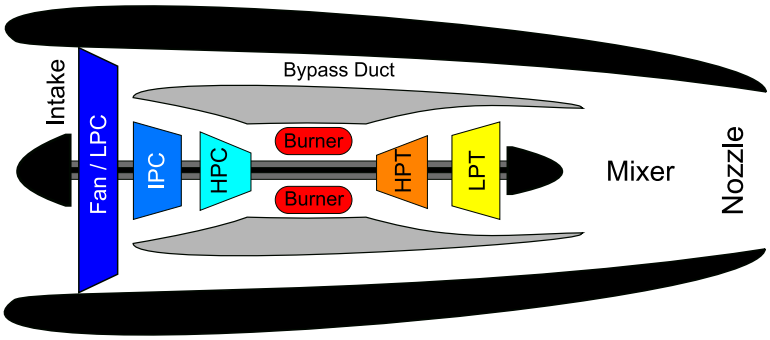


Figure 4.4: Typical configuration of a twin-spool high-bypass turbofan engine with mixed exhausts

which is concentric to the rest of the engine. In this instance, the fan essentially behaves as a shrouded propeller. A fan-exhausted, twin-spool, turbofan engine is illustrated in Fig 4.3, and is the focus of this work due to its application in large commercial airliners.

Alternatively, the core and bypass flows may be combined in a "mixer" before exiting via a single nozzle. A twin-spool, mixed exhaust, turbofan engine is illustrated in Fig. 4.4. Afterburners may be incorporated within both mixed and separate exhaust turbofan engines. The afterburners may be situated in both (or either) bypass and primary flow paths. An illustration of these configurations has not been included.

Finally, a turbofan engine may be classified as either a high or low-bypass engine. Engines with bypass flows exceeding about five times the core flow are considered to be high-bypass turbofan engines. The focus of this study is a typical twin-spool, high-bypass, fan-exhausted, turbofan engine. These engine configurations are typically used for propelling large commercial passenger aircraft.

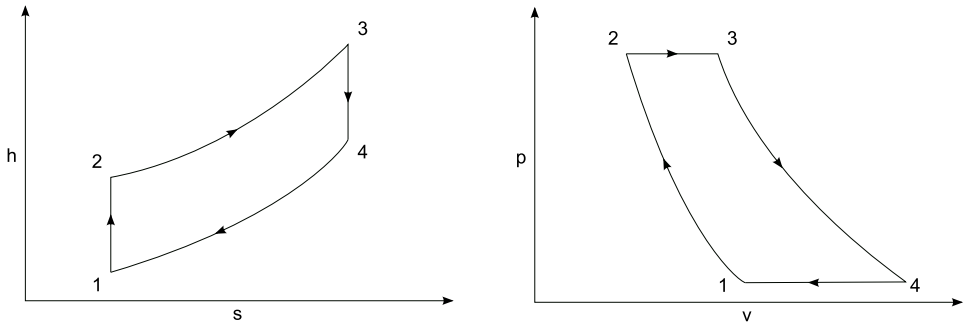


Figure 4.5: h-s and p-v diagrams of the ideal Brayton cycle

Other engine configurations, such as the turboprop and turboshaft engines, have been excluded from this section to maintain brevity.

4.1.2 Engine Thermodynamic Cycles

The ideal thermodynamic cycle of any gas turbine engine is based on the Brayton cycle, with its h-s and p-v diagrams as shown in Fig. 4.5. As is seen on the diagrams, compressors pressurise the air isentropically between points 1 and 2. Heat is then added to the working medium at constant pressure, between points 2 and 3. In an actual gas turbine engine this is done through combustion of fuel. The high energy fluid is then expanded through the turbine isentropically from point 3 to 4. Lastly, the working medium is returned to its initial state by passing it through a heat exchanger at constant pressure.

The thermal efficiency of the system is calculated from the enthalpy values at each vertex of the cycle and, by assuming isentropic compression and expansion, may be related to the compression ratio. A mathematical description of the thermal efficiency of the Brayton cycle is given in Equation 4.1.

$$\eta_{th,Brayton} = 1 - \frac{h_4 - h_1}{h_3 - h_2} = 1 - \left(\frac{P_{t,1}}{P_{t,2}}\right)^{\frac{\gamma-1}{\gamma}} \tag{4.1}$$

It is evident that increasing the pressure ratio of a gas turbine engine will in-turn increase its thermal efficiency however, the pressure ratio is restricted by the weight of additional compressor stages.

However, in an actual engine, compression and expansion will not occur isentropically, and the combustion process will inevitably suffer from pressure losses. The h-s diagram, for non-ideal operation, shifts to incorporate these effects and is illustrated in Fig. 4.6.

Furthermore, an actual engine does not operate as a closed system, as the diagram of the Brayton cycle suggests. In actuality the engine must ingest fresh air at the start of the cycle, and expel exhaust gases at the

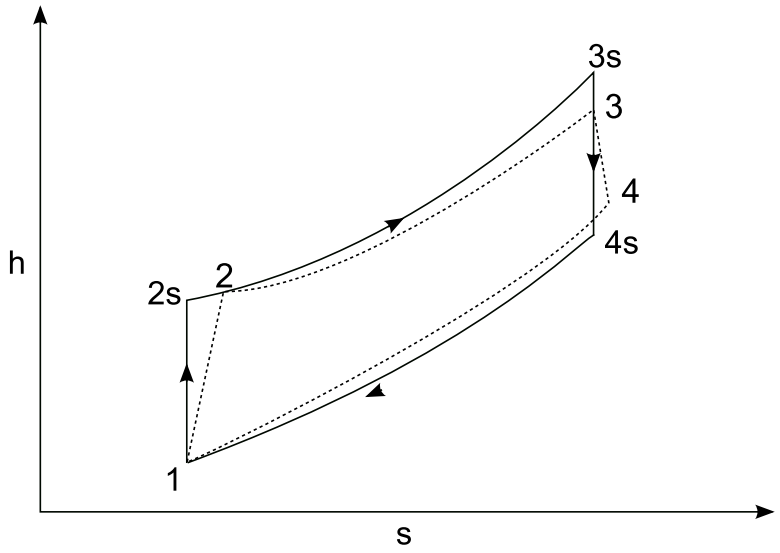


Figure 4.6: h-s diagrams of a non-ideal Brayton cycle

end of the cycle. In fact, it is the exhaust process that is responsible for the change in momentum that results in thrust. The thermodynamics of ingesting the air and expelling the exhaust gases needs to be considered too.

In the case of an single-spool turbojet engine, air is ingested by the intake and undergoes diffusion. The diffusion at the intake is there to slow down the flow before it reaches the face of the compressor. In addition, it serves to recover some of the kinetic energy in the flow, thereby increasing the total pressure entering the compressor. The diffused air enters the compressor, where it is pressurised adiabatically. During compression, work is done by the compressor on the fluid to increase its total pressure and temperature. The compressed air is passed to a combustion chamber, where fuel is injected and ignited. The combustion process occurs at a relatively constant pressure and increases the enthalpy of the gas. The high-pressure, high-temperature gas is then expanded through the turbine. During expansion, work is done by the high-energy gasses on the turbine. The power generated at the turbine is used to turn the compressor. After exiting the turbine there is still significant amounts of enthalpy available in the working fluid. This gas is expanded through a nozzle, where the remaining enthalpy is converted to kinetic energy. Thus, the remaining gases are exhausted coaxially at a much higher velocity than that of the ingested air. This change in momentum is what results in thrust.

A similar description may be used to detail the operation of a twin-spool engine, and therefore the operation of a turbofan's core. In the case of a turbofan engine, a turbine is used to drive a shrouded fan which imparts a smaller velocity change to the bypass flow in comparison to the core flow.

However, this small velocity change is applied to a larger volume of air, resulting in a large generation of thrust. Thrust is discussed in further detail in the next subsection.

4.1.3 Performance Parameters

As mentioned previously, the change in momentum of the working fluid from inlet to exit is what is responsible for thrust. Exhaust gases are expanded out of the engines nozzle, where the remaining enthalpy is converted to kinetic energy. If the exhaust pressure is expanded to equal the free-stream pressure, then the thrust is given purely by the momentum change of the working fluid. Under this ideal condition, the expansion process operates at its highest efficiency and the greatest thrust is achieved. However, there is usually a difference in pressure between the free-stream and the exhaust gases. Equation 4.3 gives the thrust produced at each nozzle of the engine. At first glance it appears as though an under-expanded nozzle is desirable, as the pressure term becomes positive, thereby seemingly contributing to the overall thrust produced. However this is not the case, as under-expansion detracts from the speed at which the exhaust gases are expelled, and this negative effect offsets the benefit of the pressure differential. In fact, there is an overall decrease in engine thrust. On the other hand, it is clear that an over-expanded nozzle is detrimental to optimizing thrust.

$$F = (\dot{m}_e U_e - \dot{m}_\infty U_\infty) + A_e \left(P_e - P_\infty\right) \tag{4.2}$$

In the case of a turbofan engine, the total thrust is obtained by summing the individual thrust contributions from the core and bypass flows, as well as the negative contribution of ram drag, Equation 4.2.

$$F_{total} = F_{core} + F_{bypass} + D_{ram} \tag{4.3}$$

Thrust is improved by imparting momentum to a small mass at high speed or a large mass at low speed. The latter is generally more efficient, and therefore a high-bypass ratio is used, typically, in commercial applications. Thrust may also be converted to a dimensionless form for ease of comparison of an engine's performance between different operating conditions (e.g. altitude, where speed of sound differs):

$$\bar{F} = \frac{F}{\dot{m}\, a} \tag{4.4}$$

Thrust is obviously an important performance measurement, because it determines whether the engine will be able to sustain the flight of a specific aircraft. Furthermore, enough thrust must be made available for take-off and in some cases extreme manoeuvring.

Engines are normally rated according to the maximum thrust they are able to produce at static sea-level conditions, and it is important to note that this value will decrease with increases in altitude and flight speed.

Although thrust is the most important performance measurement of an engine, it does not indicate the efficiency at which that thrust is produced. Therefore, the thrust specific fuel consumption (TSFC) is used for comparison between engines. It compares the amount of fuel required to produce a certain amount of thrust, and is usually defined for an engine's maximum static-thrust produced at sea level. The TSFC is simply given as:

$$TSFC = \frac{\dot{m}_f}{F} \tag{4.5}$$

The TSFC allows for easy comparison between fuel economy of different engines. Another common performance parameter is an engine's thrust-to-weight ratio. This is a very important performance parameter for aircraft because a high thrust engine is pointless if it increases the aircraft's mass by too much.

Nonetheless, these performance parameters will all vary for different flight conditions and engine operation. The following two subsections discuss such operations, namely steady-state (design and off-design) and transient operation.

4.1.4 Steady-State Operation

As mentioned earlier, it is the engine components and their layout which defines the operation of the overall engine. For the engine to operate in a state of equilibrium, a compatibility between engine components must be reached. This means that the compressor-turbine pairs must operate at the same power (i.e. work compatibility), and that continuity must be conserved across the entire engine (i.e. flow compatibility).

The design point operation of the engine is an example of steady-state operation. At this point, the engine operates at it highest efficiency. This operating point, although desired, may or may not coincide with the optimum efficiencies of each of the component's operation. Instead, it represents a point where the best combination is achieved. However, an engine does not usually get to operate at this design point, and instead operates at a less efficient (but more practical) off-design steady-state operating point.

The behaviour of each component is uniquely dependent on its boundary conditions and, in the case of compressors and turbines, its operating speed. Typically, the steady-state performance of a compressor or turbine's operation is described by performance/characteristic maps. These maps have been detailed in Appendix D, and it is recommended that the reader refers to this appendix before continuing, should they lack familiarity with such representation of turbo-machines. Therefore, steady-state operation occurs

for spool speeds and fuel flow rates at which flow and work compatibility are achieved. This steady-state operation may be calculated iteratively from mathematical descriptions of component thermodynamics and their accompanying steady-state performance characteristics. The process of finding a point of steady-state operation is referred to as "component matching", and is described in most texts on gas turbine engines.

A nested-loop approach may be used to establish equilibrium operating points. Such a method is presented by Flack [33]. In this approach, the inner loop iteratively determines the mass flow rate of a compressor-turbine pair, for a given spool speed and fuel flow rate, such that work and flow compatibility are satisfied. The outer loop compares the mass flow rate calculated in the inner loop to the flow rate permitted by the nozzle. If these do not match, a new spool speed is estimated and the inner loop is run once again. Another nested-loop type approach is presented by Sanghi *et al.* [35]. A nested loop approach soon becomes very complicated and difficult to program as the complexity of the engine increases. Furthermore, nested loops are numerically inefficient, and alternative methods are better suited to the task of component matching.

Typically, the engine's operating point is defined by a set of non-linear equations, which are formulated by the combination of mathematical descriptions of the thermodynamic operation of each of the engine's components. A set of parameters may be chosen from these equations to form a state vector, which fully defines the engine's equations. An estimate of the state vector allows for a cascading calculation of engine thermodynamics from intake to nozzle. Next, flow and work compatibility are examined and the state vector re-estimated to mitigate any torque or flow imbalances. The state-vector may be solved iteratively, using a multi-variable Newton Rhapson scheme, for example, to ensure that flow and work compatibility are reached. Such a method is described in detail by McKinney [36, 37], along with documented FORTRAN code and a user manual. This type of approach easily accommodates engine complexities, as more parameters may be introduced within the state vector.

Nonetheless, the path connecting steady-state operating points, of highest efficiency at each specific engine speed within the engine's operating envelope, forms what is referred to as the engine's "running line". Its is obvious that, should an engine be required to accelerate/decelerate, it is desirable that it adhere to its running line as closely as possible. However, this is usually not practically possible, and the engine will inevitably deviate from the running line into regions of lowered efficiency. It is therefore evident that there is value in being able to predict transient paths, when analysing engine performance. The next subsection briefly describes engine dynamic operation, before moving on to a literature review into the many ways in which engine dynamics have been modelled.

4.1.5 Dynamic Operation

Engine transients are encountered when an engine is perturbed from a point of equilibrium; where work and/or flow compatibility are disrupted. The engine will attempt to regain a point of equilibrium (when it is in a stable mode of operation) that once again satisfies work and flow compatibility. However it is physically impossible for this to occur instantaneously, and therefore time is taken to move from one operating point to the next.

Most commonly, it is the change in fuel flow rate that initiates engine transients. Although it must be noted, that fuel flow is the actuation used to control, usually, the engine's turbine inlet temperature. When the fuel flow rate to the combustion chamber/s is increased, energy is added to the system. This additional fuel flow leads to increases in turbine inlet temperature, resulting in increased turbine power. The additional power will surpass the spool inertias, as well as energy demands of the compressor, causing the spool to accelerate. The higher spool speed increases the operating pressure ratio of the compressor, and a larger mass flow rate is ingested. The pressure and temperature to the turbine is increased further, resulting in continued acceleration. This is repeated until the engine reaches a new point of equilibrium, where the additional power just sustains the new energy demands of the compressor and the spool inertia at the new rotational speed. The opposite will occur when the engine decelerates.

During acceleration/deceleration, the engine components are subjected to various conditions which move their operation away from their design point. It is important to be able to predict the transient path of the the engine and its components, in order to understand any impacts on performance. Simulations are required to aid the design of engine hardware and control systems. In addition, simulations may be used to monitor the health of an engine as well as predict performance degradation. Moreover, as is the case of this work, dynamic engine simulations are often used in conjunction with aircraft simulators to better understand the performance of the entire aircraft in various dynamic scenarios. The next section discusses the many methods of simulating engine transients that have been developed since the jet engine was first conceived.

4.2 Evolution of Transient Simulations

Overviews of the literature regarding the simulation of gas turbine dynamics are presented by Saravanamuttoo and MacIsaac [38], Evans [39] and Sanghi *et al.* [40]. Saravanamuttoo and MacIssac's overview contains a description of modelling requirements and basics, as well as an evolution of literature spanning from the 1950's to the early 1980's. Evans provides a brief introduction to gas turbine systems and lists some of the prominent works in gas turbine simulation, up until the mid 1990's, with particular attention

paid to the effects of heat transfer. The literature review by Sanghi *et al.* contains a comprehensive list of gas turbine theoretical work up until the turn of the century. Sanghi *et al.* cover engine fundamentals, steady-state modelling, and dynamic simulations ranging from high-fidelity models to real-time simulations. These sources were used as a starting point for uncovering the literature regarding the simulation of gas turbine dynamics. The subsections that follow, track the advancements made in simulating engine dynamics, from the late 1940's right through to the latest literature available today. Also detailed, are some of the many commercial software packages that are now available for use by the public. Finally, the methods found in the literature are discussed in context to their applicability in this work. This leads on to the next chapter which presents the engine model built for investigating the fuel consumption in formation flight.

4.2.1 Early Linearised Approximations

Advancements in gas-turbine engine technology began to accelerate toward the end of the second world war, and with it the requirement for improved engine design and control. It therefore became necessary to understand and predict engine dynamics before, during and after the design stage. When system dynamics are quite complex, as is the case with gas turbine engines, it is often convenient to derive approximate mathematical models from experimental data. Such methods, applied to gas turbines, were first introduced in 1949 by LaVerne and Boksenbom [41], who used the response of a turboprop engine, to changes in propeller-blade-angle, as an example of its application.

Later that year, Boksenbom and Hood [42] addressed, in detail, the issues of controlling gas turbine transients. A generalised, control-systems approach was undertaken to accommodate the numerous types of gas-turbines undergoing development at the time. This approach recognised the necessity of multi-variable control, for the safe operation of the engine; preventing excessive temperatures, speeds, torques and combustion flame-out. The method attempted to eliminate any unwanted interactions between the various control variables (both dependent and independent), as well as adhering to design requirements of response time, under/over-shoot and steady-state tracking error.

Boksenbom and Hood reasoned that engine transients could be predicted analytically, assuming linearity for small perturbations around a known steady-state operating condition. The transient model is defined by a set of dependent and independent variables, according to the purpose of the control system under analysis or design. Assuming linearity, each dependent variable may be expressed as summation of sub-functions relating it to each independent variable, as indicated in Equation 4.6:

$$y_j = \sum_{k=1}^{n} E_{jk}\, x_k \quad j = 1, 2, \ldots m \leq n \tag{4.6}$$

where, x and y are the independent and dependent variables of size m and n, respectively. The sub-functions, denoted by E, relate a set of dependent and independent variables, and are determined from engine test data. In addition, Boksenbom and Hood provide an example of their approach applied to a theoretical turboprop engine.

In 1950 Otto and Taylor [43], of the National Advisory Committee for Aeronautics (NACA) in the US, attempted to approximate the transient behaviour of a single-spool, turbojet engine using analytical methods. A differential equation was formulated based on spool accelerations due to the torque imbalance between the compressor and turbine. This relationship is given by Equation 4.7, where the difference in torque was assumed to be function of fuel flow, combustion efficiency and spool speed; Equation 4.8.

$$\dot{N} = \frac{Q}{I} \tag{4.7}$$

$$Q = f\left\langle \dot{m}_f, \eta_b, N \right\rangle \tag{4.8}$$

By assuming a quasi-static system, and hence ignoring flow and thermodynamic mismatches between components, Equation 4.8 was expressed in terms of its partial derivatives via Taylor expansion. Assuming small perturbations about a steady-state operating condition, the high-order terms are ignored and the equation linearised. Furthermore, because a quasi-static system was assumed, the torque imbalance during steady-state operation is zero. Hence Equation 4.8 may be expanded and simplified to yield Equation 4.9.

$$Q = \left[\frac{\partial Q}{\partial\left(\dot{m}_f \eta_b\right)}\right]_0 \Delta\left(\dot{m}\eta_b\right) + \left[\frac{\partial Q}{\partial N}\right]_0 \Delta N \tag{4.9}$$

Combining equations 4.7 and 4.9 produces a linearised, first-order, differential equation that models engine acceleration as a function of fuel flow and running speed:

$$I\frac{d\left(\Delta N\right)}{dt} = \left[\frac{\partial Q}{\partial\left(\dot{m}_f \eta_b\right)}\right]_0 \Delta\left(\dot{m}_f \eta_b\right) + \left[\frac{\partial Q}{\partial N}\right]_0 \Delta N \tag{4.10}$$

The terms in Equation 4.10 must then be corrected for altitude, using standard methods, which allows for the equations to be used at different flight conditions. The coefficients of the resultant differential equation may be obtained using experimental data of the engine's response to step inputs of the independent variables. Alternatively, these values may be obtained using steady-state component characteristics. Otto and Taylor demonstrated

how the difference in torque may be calculated for variations in assumed values of engine speed, turbine inlet-temperature and ram effects. Hence the engine acceleration may be plotted as a function of running speed and fuel flow (indicative of turbine inlet temperature), and their slopes used to determine the coefficients for Equation 4.10. The result is a general, approximate equation representing engine dynamics, which may be used to predict engine performance for a range of flight conditions.

Unfortunately this method only demonstrated good correlations, against experimental data, for small accelerations about a known steady-state operating point (as is to be expected). This was attributed to the observation of large variations in engine time constants between steady-state operating points, as well as deviations in combustion efficiency during transients. Furthermore, greater errors were observed during deceleration rather than acceleration of the engine, which was attributed to reduced combustion efficiencies.

Taylor and Oppenheimer [44] extended this method to model the transients of a turboprop engine by incorporating propeller-blade-angle as one of the independent variables. The purpose of this investigation was to compare the experimental results of engine frequency-response against those predicted by the linearised, first-order approximations. In so doing, the engine speed was investigated for oscillatory inputs of fuel flow and propeller-blade-angle. The results highlighted higher-order effects inherent in the engine dynamics. Furthermore, linear approximations only yielded good correlations for spool speed deviations, from a steady-state operating condition, of less than 3-4%. In addition, it was observed that correlations worsened for increases in frequency and/or phase changes. This was because the higher order effects become amplified in these operating conditions. Lastly, it was also noted that engine speed time constants were significantly different from one steady-state operating point to the next.

Himmel and Krebs [45] adopted a similar method for analysing changes in transient characteristics of a turboprop engine at different altitudes. The purpose of the investigation was to examine the changes in control required at different altitudes.

Dandois and Novik [46], in 1952, tested the adequacy of such linear approximations for engine control purposes. Curves of fuel flow, air flow and turbine exit temperature against engine running speed were obtained from steady-state experimental data. These curves were used to formulate a first-order, linearised, transient, analytical model of the engine. The time constants were obtained at different steady-state operating points. As with previous studies, these time constants were observed to vary significantly for different engine speeds, as illustrated by Figure 4.7.

This analytical model was compared to experimental data for open-loop step and frequency responses to changes in fuel flow. Fair correlations were achieved for these responses, however non-linearities became more visible at

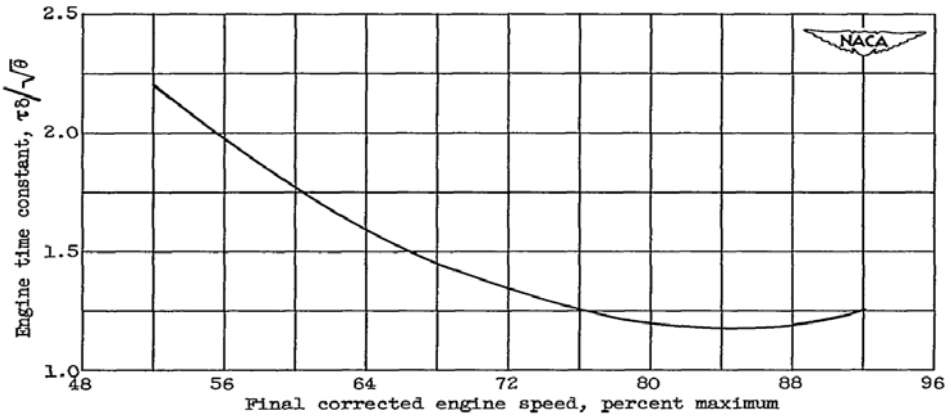


Figure 4.7: Variations in first-order time-constants with engine running speed. Reprinted from Dandois and Novik [46]

higher frequencies. Nevertheless, a proportional speed controller, actuating fuel pump pressure, was included and the stability of the system examined. Closed-loop responses highlighted non-linearities existing within the system, and a second-order model was used to achieve a better correlation to experimental results. It was concluded that although the linearised model gave reasonable correlations for open-loop responses, a higher-order model is required to investigate engine stability for closed-loop responses.

Due to inherent non-linearities present in gas-turbine engine dynamics, the linear approximations, presented thus far, were not suitable for predicting transients of large accelerations. An experimental examination of large turbojet accelerations (open-loop step-response to fuel input) was conducted at the Lewis Flight Propulsion Laboratory in 1952 [47]. The results revealed that accelerations from a steady-state operating point, to engine-rated-speed, were only linear for differences of less than 15% between initial and final engine speeds (see Figure 4.8). Nevertheless, it was noted that this region of linearity was sufficiently large to aid the control analysis of most manoeuvres that may be required of an aircraft. Other observations of the experiments highlighted the importance of fuel metering for improvements in acceleration characteristics, which was attributed to lower the higher order effects of combustion and compressor surge.

Although first-order approximations did not allow for accurate predictions of large accelerations, it did provide a convenient tool for small perturbation and stability studies of engine dynamics. Furthermore, they were well suited to the computational capabilities available at the time and therefore warranted further development.

In 1952, Ketchum and Craig [48] generalised first-order methods to incorporate any number of dependent and independent variables that may have been deemed necessary, or were required to describe a particular en-

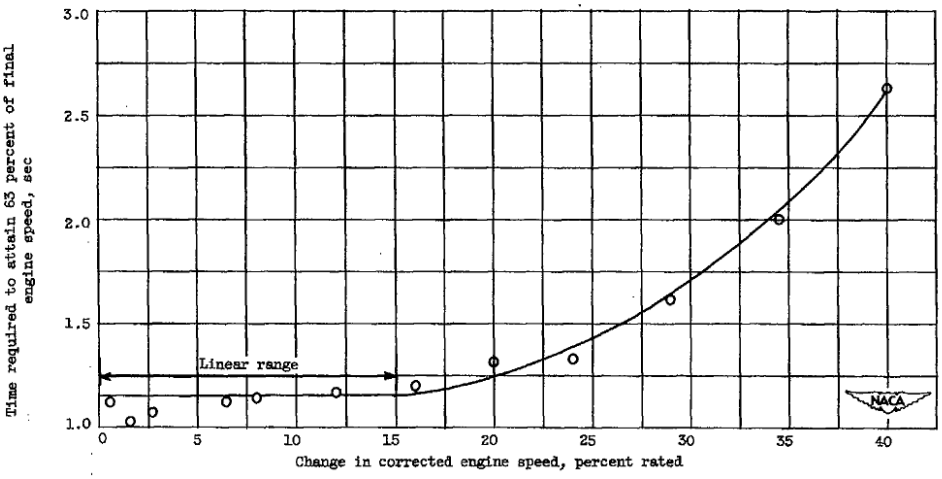


Figure 4.8: Time constants for accelerations to rated engine speed, illustrating a region of linearity of 15% of rated speed. Reprinted from Heppler, Novik and Dandois [47]

gine type. In addition, the generalised approach took into careful consideration the impact that variations in altitude and flight speed may have on the fidelity of transient predictions. This generalised method was used to examine transients of both a turboprop and turbojet engine. Once again, engine transients are assumed to be quasi-static and the set of equations are expanded and linearised, as before, using Taylor expansion. The time constant, for the resulting differential equation, was determined from transient test data and the remaining coefficients from steady-state data. The simulations were performed using an analog computer and comparisons were made against transient engine test data. The method yielded good correlations with test data for step responses about a steady-state operating point, see Figure 4.9.

Although the linearisation of engine transients does not allow for accurate predictions of large accelerations, these simple models provided a fast and convenient tool for designing and analysing control systems. Pack and Phillips [49], for example, make use of Ketchum and Craig's simulation methodology to investigate engine control.

Twin-spool engines were born out of the necessity to increase compression ratios for improved engine performance. As the complexity of engines increased so did the requirement for simulations that would aid the design of engine hardware and its control systems. In 1954, Novik and Heppler [50] analysed the transient response of two twin-spool turbojet engines, each built by a different manufacturer. Similar to before, the dynamics of each spool was modelled using a linearised, first-order differential equation that

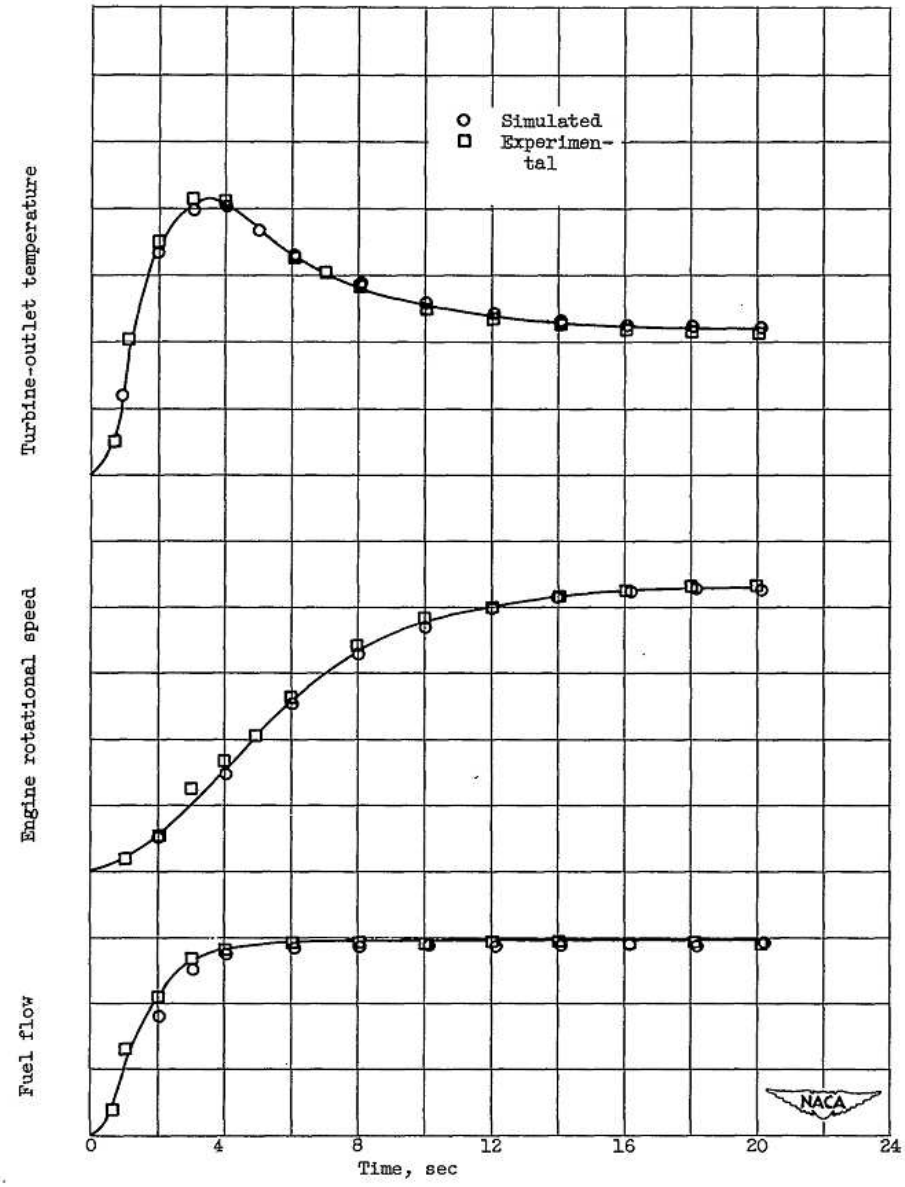


Figure 4.9: A comparison of first-order turbojet transient simulation response to that of engine experimental data, conducted by Ketchum and Craig [48]

correlated mismatches in torque to spool accelerations. The torque imbalance, for each spool, was characterised as a function of both spool speeds and turbine inlet temperature (indicative of fuel flow) as shown in Equations 4.11.

$$I_{HP}\frac{d\left(\Delta N_{HP}\right)}{dt} = \frac{\partial Q_{HP}}{\partial N_{HP}}\Delta N_{HP} + \frac{\partial Q_{HP}}{\partial N_{LP}}\Delta N_{LP} + \frac{\partial Q_{HP}}{\partial T_{i,HPT}}\Delta T_{i,HPT}, \tag{4.11a}$$

$$I_{LP}\frac{d\left(\Delta N_{LP}\right)}{dt} = \frac{\partial Q_{LP}}{\partial N_{HP}}\Delta N_{HP} + \frac{\partial Q_{LP}}{\partial N_{LP}}\Delta N_{LP} + \frac{\partial Q_{LP}}{\partial T_{i,HPT}}\Delta T_{i,HPT} \tag{4.11b}$$

The above equations were used to examine responses for changes in fuel flow and exhaust nozzle area. From experimental step responses, it was observed that both spools act as first order systems with equivalent time constants. Therefore, theoretical transfer functions could be obtained based on these data. However, discrepancies in frequency responses occurred between test and simulated results, with experimental results indicating responses of a higher order. This was attributed to lag times that may occur in the combustion process, thereby delaying the change in turbine inlet temperature. Nevertheless, a good correlation between theoretical and experimental results was achieved, but once again it must be noted that this method only provided good correlations for deviations close to a predefined steady-state operating position.

In summary, linear approximations may provide some limited insight into the behaviour of gas turbine dynamics, and were convenient for designing and analysing early control systems. However, linear approximations are unable to represent the dynamics of large disturbances. Furthermore, a set of partial derivatives and time constants need to be determined for different steady-state conditions. Piecewise linear approximations have been suggested as a method to curb such limitations [51]. Nonetheless, such an exercise would be time consuming, difficult and impractical. Therefore thermodynamicists began looking for ways to model the non-linear dynamics of gas turbines.

4.2.2 Non-Linear Simulations

The linearised models referred to thus far, were highly reliant on transient and/or steady-state test data for establishing rotor time constants. Furthermore, linear approximations were simply unable to give accurate predictions for large accelerations and were also unable to predict performance over the entire operating range of the engine. Therefore a new simulation method, that could be obtained from component steady-state data and accounted for non-linearities, was required.

Early Non-Linear Models

The modelling of non-linear dynamics was approached from a thermodynamic point of view. At this stage the thermodynamics of each engine component, could be described accurately by mathematical formulation using component steady-state characteristics. Therefore, the thermodynamics of the entire engine were simply defined by the layout of these components and their thermodynamic interdependencies. The next step was to determine how to mathematically represent the interdependent operation of components.

Dugan, in 1954 [52], introduced component matching procedures for twin-spool engines, which could be used to aid predictions of transient engine performance. In this procedure, the components coupled to the high-pressure spool are matched first, and then the components coupled to the low-pressure spool are matched to operation of the high pressure spool. Spool component matching is based on three simple principles. Firstly, the mass flow rate through the turbine/s must equal that of the compressors (accounting for any bleed accordingly) plus that of the fuel added. Secondly, compressors and turbines attached to the same spool rotate at the same speeds. Thirdly, any power generated by the turbine, and not used to power auxiliaries, is fully absorbed by the compressors on the same spool. Once the two gas generators had been matched, the operating points of the other engine components could be calculated. Dugan made use of graphical methods to manually complete the matching procedure from steady-state operating curves of the various components.

The above matching procedure aided the prediction of engine non-linear transients. Following a disturbance from an equilibrium operating point (e.g. increase/decrease in fuel flow), the engine is matched according to the first two criteria only; continuity and speed. This results in a power imbalance for each spool, from which their respective accelerations are calculated. Assuming engine conditions are held constant momentarily, new spool speeds are determined at the end of the time step. At this new running speed, the continuity condition will no longer be satisfied and the matching procedure must be conducted again, hence finding a new power imbalance. This is repeated until the engine reaches a new point of equilibrium.

This technique was used to examine the operational trends of compressor and turbine components during engine accelerations for different divisions of spool inertias [53] and compression ratios [54] between spools. The focus of such work was to investigate compressor surge and spool over-speeding during engine design. Although this method described engine dynamics more accurately, it was not practical for many applications because of the tedious manual calculations and graphical work required for component matching. Nonetheless, it would form the basis for future work on non-linear models, with computers automating the tedious manual calculations.

Non-Linear Analog Computations

In the late 1950's, analog computers were available to perform engine simulations. Larrowe and Spencer were responsible for much of the early development of analog simulation of gas turbine engines [55–57]. Larrowe and Spencer made use of standard analog computing equipment, with the addition of a modified electro-mechanical X-Y plotter, which was used to read component characteristic maps. DC voltages were used to represent engine variables; rotor speeds, enthalpies, pressure ratios and flow rates. Any type of engine could be simulated by replicating the configuration of the the engine's components. Therefore, engine dynamic behaviour could be investigated before it was built.

The analog computer described above, calculates engine non-linear dynamics from a thermodynamic perspective. A flow chart of the logic used is illustrated in Figure 4.10, with station numbering representing the inlets and outlets of the components, as indicated on the engine schematic. The engine variables all begin at some initial value, usually the steady-state design point, where the engine is at equilibrium. Following a disturbance from the initial conditions, the engine transients can be determined for incremental time-steps. First, Changes in mass flow rate and enthalpy are cascaded downstream from the compressor, and then effects on pressures are propagated upstream from the nozzle. The resulting power imbalance between the compressor and the turbine is then used to calculate the spool's acceleration over a given time-step.

Starting at the compressor, the inlet flow rate and enthalpy rise is determined from the pressure ratio across the compressor and the spool speed. These functions are not easily described mathematically, and therefore compressor characteristic maps are used instead. Such maps were read using a modified electro-mechanical X-Y plotter. For this example there are no compressor bleeds, and therefore the outlet mass flow rate is equal to the inlet flow rate. The inlet conditions to the combustor are now defined. The fuel mass-flow-rate is added to the inlet flow, and a rise in enthalpy is given as a function of the air-fuel mixture. Hence, the mass flow rate and temperature inlet conditions to the turbine are altered. Like with the compressor, the turbine outlet conditions are determined from characteristic curves, thus defining the inlet to the nozzle. Given the ambient pressure and nozzle area, the pressure ratio across the nozzle is calculated, and the thrust determined. The nozzle pressure ratio now defines the modified outlet pressure from which the modified turbine inlet pressure is determined. This in-turn defines the combustor outlet pressure which may be used to calculate its inlet pressure and the exit pressure of the compressor. Next the compressor's pressure ratio is determined from its characteristic curves. Finally, with the thermodynamic modifications to the engine calculated, the power imbalance between compressor and turbine may be integrated to de-

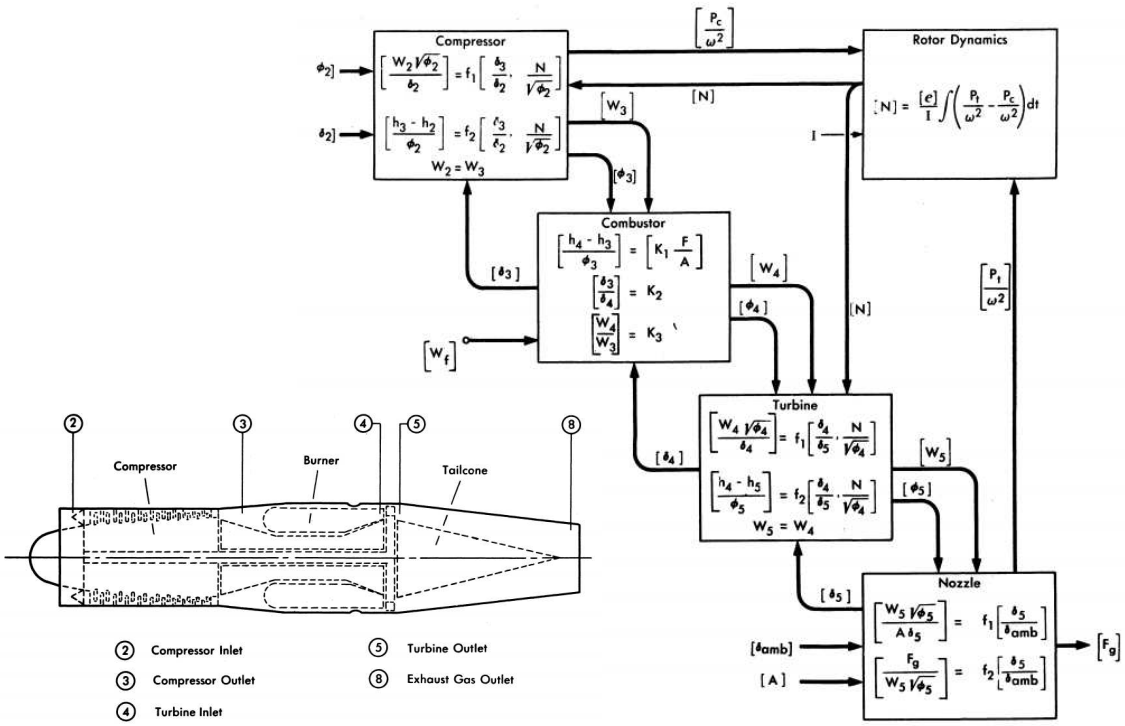


Figure 4.10: Flow chart of the analog computer calculations of a simple single spool turbojet engine. Reprinted from Larrowe and Spencer [55]

termine the spool's acceleration for a given time step. The entire process repeats itself until the engine establishes a new equilibrium operating point.

This analog methodology was able to facilitate many applications. The computer could be used to determined off-design steady-state operating conditions of the engine, investigate surge and bleed systems, analyse variable sensitivity for control studies as well as being incorporated with aircraft simulations.

Analog computations successfully carried on throughout the 1960's, with significant contributions from Saravanamuttoo [58], who improved the analog representation of component characteristic curves. The component characteristics are inherently functions of two variables. Saravanamuttoo was able to represent these characteristics using three functions of one variable, which enabled the use of reliant and accurate diode function generators. Thus curbing the inconvenience of the elaborate map readers developed by Larrowe and Spencer.

The analog simulation technique was revisited by Fawke and Saravanamuttoo [59] and improved to be more reliable, repeatable and flexible. The analog simulator was first used to test the static sea level steady-state operation of a twin-spool turbojet engine with variable nozzle area. Comparisons

with test bed experiments proved the accuracy and repeatability of the simulator. Next, the simulator was used to predict the transient operation of the same engine. The aim of this investigation was to improve the engine thrust response time by using a combination of throttle and nozzle area dynamics. The simulator was able to project transient paths onto compressor characteristic maps, using an X-Y plotter, which were used to study the transient effects on compressor surge. The results produced by the simulator were said to be proven by experimental tests conducted at the National Gas Turbine Establishment.

Analog computation was excellent for gas-turbine simulation. Real-time simulations, important for control system development, were easily achievable no matter the complexity of the engine. Analog components and signals, which were analogous to actual engine components and variables, allowed for simulators that could truly capture the physics of engine transients. This is because analog computations effectively run in parallel, thus closely mimicking the actual interactions of an engine component and thereby producing relatively accurate predictions of non-linearities.

Unfortunately, analog computers are time consuming to set-up (particular the function generators representing component characteristics), lack storage capacity and do not allow for easy adaptability to changes in configurations or component characteristics. Because of these disadvantages, engineers began to turn towards the emerging technology of digital computing in the early 1970's. Initially hybrid computers were used to combine the high-speed parallel computing of the analog with the storage capacity, flexibility and convenience of the digital computer. However the shift to solely digitally based engine models was swift, and therefore a discussion of hybrid simulations has not been included.

Non-Linear Digital Computations

Gas turbine dynamic simulations started to move toward digital computing in the early 1970's. Digital simulations became an attractive option because of the reliability and repeatability attainable. This was reinforced by large storage capacities and a high level of flexibility. Nevertheless, digital simulations faced a few challenges because of its serial based computing methodology. At the time digital computers could not perform parallel calculations, which was the inherent property of analog computers that allowed for fast and accurate computations. Furthermore, serial computing was comparatively very slow, thereby not allowing for the real-time simulation of complex engine types. Nevertheless, digital computing showed great promise and was therefore pursued.

Saravanamuttoo and Fawke were among the early pioneers to digitally simulate gas turbine dynamics [51, 59]. For a known flight condition, fuel flow rate and exhaust nozzle area, it was proposed that an engine's operating

point could be fully defined by a state vector of certain engine parameters. These parameters included the exit pressures of all rotary components and the operating speed of each spool. Assigning values to these parameters would define the mismatches in torque, between corresponding compressors and turbines, as well as mismatches in the flow rate between adjacent components. Flow and work compatibility could be dealt with in one of two ways.

The first method assumes that flow compatibility is maintained throughout the engine at all times. At each time step, the components are matched thermodynamically for a given spool speed, using iterative methods, thus leaving only an imbalance between compressor and turbine work. The difference in torque is used then to calculate the acceleration of each spool based on its inertial characteristics, as represented by Equation 4.7. The acceleration is integrated to find the new speed of each spool at the end of the time step. Therefore, a new set of values is assigned to the state vector, and the process is repeated until equilibrium is achieved once again. A change in external condition (e.g. fuel flow, altitude or flight speed) will result in a new imbalance and the process will begin all over again.

The second method allows for mass accumulation of flow between components. This mass accumulation is used to calculate the rate of change of the state vector's pressures. Each component will require a certain mass flow rate (according to its operating characteristics) for a given running speed an pressure ratio (defined by the state vector). When the engine is not at equilibrium, there will exist a discrepancy between the mass flow rates required between adjacent components. This discrepancy is accounted for by assuming that mass is accumulated in the volumes located between adjacent components. The rate of change in pressure inside these volumes is obtained from the derivative of the ideal gas law, as per Equation 4.12.

$$\dot{P} = \frac{R}{V}\frac{d}{dt}(mT) = \frac{R}{V}(\dot{m}T + m\dot{T}) \tag{4.12}$$

Assuming that $m\dot{T}$ is relatively small in comparison to $\dot{m}T$, the pressure derivative becomes a function of the rate of change of mass in the volume, which is given by the difference between the mass flow rates demanded by adjacent components; Equation 4.13. Therefore, the rate of change of pressures and spool speeds are both accounted for and may be integrated to obtain the state vector at the next time step.

$$\dot{P} = \frac{R}{V}(\dot{m}_i - \dot{m}_e)T = \frac{R}{V}\Delta\dot{m}T \tag{4.13}$$

Fawke and Saravanamuttoo refer to these methods as the iterative and inter-component volume method (ICV), respectively. The ICV method is more physically representative of engine dynamics as it takes into account

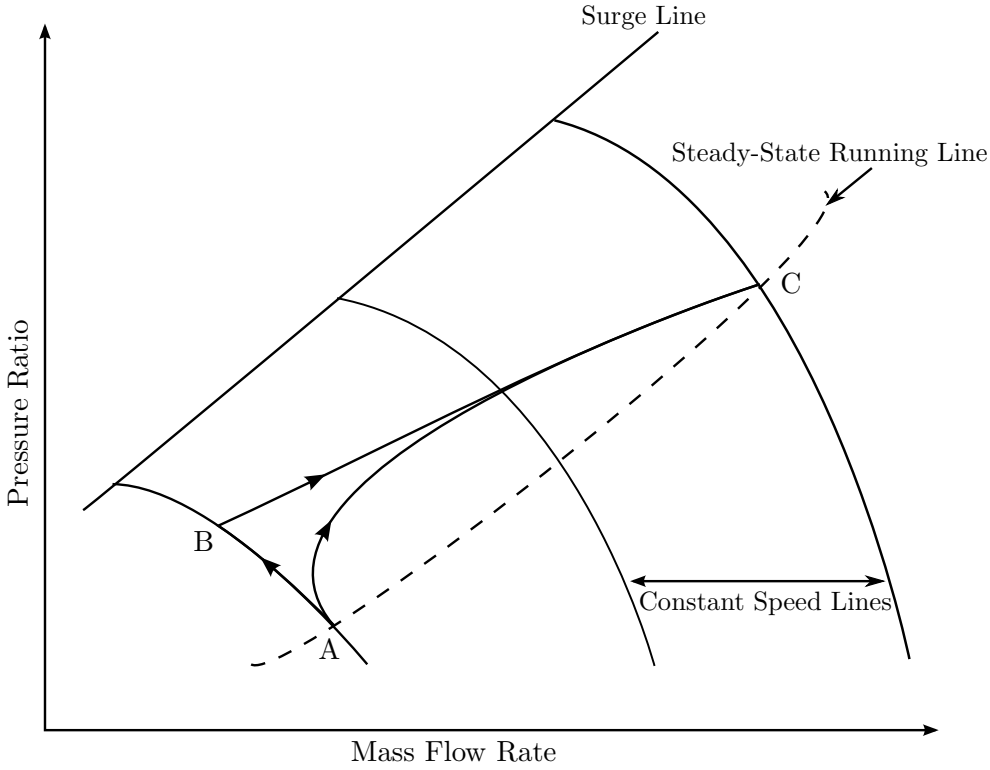


Figure 4.11: Comparison of compressor acceleration trajectories between the Iterative Method (ABC) and the Inter-Component Volume Method (AC). Reproduced from Fawke and Saravanamuttoo [51]

pressure/volume dynamics. The iterative method, on the other hand, assumes that pressures and mass flow rates reach equilibrium instantaneously, which is not physically possible. Therefore the two methods yield different trajectories on the component characteristic maps, with discontinuities present in the iterative method. The difference between the two methods is best indicated by the schematic of compressor trajectories, as shown in Figure 4.11.

As shown by the acceleration trajectories in Figure 4.11, the Iterative Method moves from position A to B and then to C. A new pressure ratio and mass flow rate are first determined for a given spool speed. The resulting work imbalance results in acceleration of the spool, and the path moves from position B to C. The ICV Method, on the other hand, moves smoothly from position A to C, simultaneously accounting for changes in pressure, mass flow rate and spool speed. It is evident from this diagram that the inter-component volume method is more representative of the actual engine dynamics. This diagram also gives an indication that the ICV Method is more suitable for predicting large accelerations, particularly when surge is of

concern. The iterative method is more likely to cross the surge line because it moves directly up the constant speed line. The ICV Method allows for larger accelerations as it does not over-predict surge. One may argue, in this instance, that the iterative method deals with surge more conservatively, and therefore is more appropriate for certain applications.

Other observations evident from Figure 4.11 are those regarding computational speed. The Iterative Method's transient path (BC) is more linear than that of the ICV Method (AC), thereby allowing the Iterative Method larger time steps. Furthermore, the equalisation of pressures and mass flow rates, during the ICV Method, occur quickly within the engine, and these fast dynamics require small time steps. In fact, a mathematically stiff problem results because of a combination of comparatively fast volume and slow rotor dynamics. However, the gains in speed achieved by the Iterative Method, in utilizing larger time steps, may be mitigated by the time taken for the iterative calculations to move the start of the transient from point A to B. Fawke and Saravanamuttoo [51] therefore recommend the use of the Iterative Method for gradual transients and the ICV Method for large accelerations over short periods.

Fawke and Saravanamuttoo placed emphasis on the use of the ICV Method. They used this method to predict the transient performance of various types of engines including; single-spool and twin-spool turbojets, as well as a twin-spool turbofan engine with mixed exhausts. The results revealed acceptable correlations to experimental test data for both steady-state operating points and engine transients. Observations of the comparisons revealed that accuracy was insensitive to large deviations in the values used to represent the volumes between components. Therefore, volumes larger than the actual physical values could be used to increase time steps, thereby improving computational speed. However, this was not true for the turbofan engine with mixed exhausts, and discrepancies between simulation and test results occurred. The reasoning for this discrepancy was attributed to the bypass duct pressure ratio, which dictates the split in flow (bypass ratio) directly behind the fan. Although not explicitly mentioned, it is assumed that this reasoning also applies to a fan-exhausted turbofan engine. Such errors were amended by utilising the actual bypass duct volume in the simulation. Fawke and Saravanamuttoo [51] also included the effects of heat transfer, which resulted in improved fidelity of the simulated results. However, their heat transfer methodology was not detailed, and it was mentioned that crude assumptions were made to obtain heat transfer coefficients.

Fawke, Saravanamuttoo and Holmes [60] conducted more experimental tests to further verify the validity of the ICV Method. A fair amount of attention was paid to selecting the size of the control volumes used in determining the pressure derivatives. It was noted that this method made the assumption that flow accumulation only occurs in the volume located between two adjacent components. However, in reality, this is not true as

flow accumulation will naturally be distributed across each of the components, and therefore over the entire engine itself. It was suggested that the ICVs be given larger volumes to account for this effect. Furthermore, more insight was given to the elimination of the $m\dot{T}$ term in Equation 4.12. It was explained that the rate in temperature rise could not be analytically determined at a particular time step and required an iterative method to predict its value based on passed values. However, it was also stated that this term was less than 20% of the $\dot{m}T$ term (although no explanation was given as to how this was calculated), and could therefore be assumed zero if the volume size was enlarged to account for its effect. This was justified because the engine performance remained insensitive for large ranges in increased volume size. From experimental results it was concluded that volumes up to 2 or 3 times that actual inter-component volume could be used in the simulation, whilst still maintaining a high level of accuracy. Hence, enlarging the inter-component volumes, theoretically, led to improvements in the physical representation of engine dynamics, as well as allowing for larger time steps to be used (which improved simulation speed).

In the same study, it was noted that when transients encountered flat regions of the speed lines in the compressor characteristic maps (usually close to surge conditions and at low speeds) smaller time steps would be required which increases computation time. This is because a small change in pressure ratio corresponds to a large change in mass flow rate which in-turn results in large pressure derivatives requiring small time steps for integration. This is because of the fast pressure dynamics in relation to the slower spool dynamics. Initially a fixed step fourth-order Runge-Kutta solver was used, but in light of the above argument, speed-ups were achieved by using a variable step Runge-Kutta-Merson solver. The variable step solver allows for larger time steps to be used in the regions where constant speed-line gradients are steeper.

In 1975, a slightly different method for predicting transients was presented by Sellers and Daniele [61]. DYNGEN was an extension of GENENG [62] and GENENG II [63], which was used to calculate the off-design steady-state operation of various gas turbine configurations. DYNGEN, like the methods presented by Fawke and Saravanamuttoo, also made use of control volumes between components to account for mass accumulation, but with the addition of energy storage. Therefore, differential equations were developed based on power imbalance (describing rotor dynamics) and the continuity and energy equations (describing flow dynamics). The addition of the energy equation included terms that accounted for energy storage due to mass accumulation as well as the rate of change of internal energy in the volume. From the first law of thermodynamics, the rate of change of energy inside the control volume is given by:

$$\frac{d}{dt}E = (\dot{m}_e h_e - \dot{m}_i h_i) \tag{4.14}$$

Using the relationship between enthalpy and internal energy, and assuming pressure and internal energy are constant across the volume, the following is obtained:

$$(\dot{m}_e h_e - \dot{m}_i h_i) = (\dot{m}_e - \dot{m}_i)e + (\dot{m}_e - \dot{m}_i)PV \tag{4.15}$$

From the ideal gas law:

$$(\dot{m}_e - \dot{m}_i) = \frac{\dot{h} - \dot{e}}{RT} \tag{4.16}$$

And assuming temperature is constant i.e. $\dot{h} = c_p\dot{T} = 0$, Equation's 4.15 and 4.16 may be combined to yield Equation 4.17.

$$(\dot{m}_e h_e - \dot{m}_i h_i) = (\dot{m}_e - \dot{m}_i)e - \frac{PV}{RT}\dot{e} \tag{4.17}$$

Values for the derivatives, $\dot{P}$ and $\dot{e}$, are found using iterative procedures based on backward difference methods. Therefore the enthalpy and mass flow rate exiting the control volume may be obtained from Equations 4.13 and 4.17. The pressure ratio and efficiency of the following component are then determined, using component maps, according to these inlet conditions. This process cascades down the engine from intake to exhaust nozzle. This methodology was used in the formulation of NASA's DYNGEN software, written in FORTRAN, as an extension of the steady-state software GENENG and GENENG II. It is claimed that this technique allows for the use of larger time steps, which more than offsets the negative effects of additional iterations on computational speed. Sellers and Daniele also include, in their paper, methods of scaling component maps. Component map scaling allows for the modification of known performance maps, to components for which these characteristics are unknown.

DYNGEN's successor came about in 1983, and was named DIGTEM for Digital Turbofan Engine Model [64]. DIGTEM was a digital version of HYDES [65], which was a simulation program on a hybrid computer that could predict both off-design steady-state and transient engine performance. DIGTEM was based around a single engine configuration, comprising of two spools, bypass flow and mixed exhausts. This complex configuration could be manipulated to simulate simpler configurations such as the turbojet engine. This was unlike DYNGEN which contained many fixed variations in engine configuration. It was therefore claimed that DIGTEM allowed for greater flexibility when modelling arbitrary configurations.

DIGTEM also had improved fidelity in the inclusion of the momentum equation for modelling the flow through the bypass duct and mixed flow

augmenter. The state vector used in solving engine dynamics included; spool speeds and inter-component volume mass flow rates, as well as temperatures, which resulted in 16 state variables for the default configuration.

As with DYNGEN, component performance maps were scaled from generalised maps to meet engine design point values. However, DIGTEM included a trimming capability which mitigated engine flow and rotor imbalances, by including correction coefficients, that may occur with given design or off-design steady-state data. This allowed for exact representation of real engine data and improved steady-state fidelity. The correction coefficients are included in the transient simulations to further improve accuracy. These correction coefficients could be calculated for any off-design steady-state operating point, in addition to the design point values. Furthermore, DIGTEM made use of a Newton-Raphson backward difference method of integration instead of a Euler forward difference method. This was done because the engine dynamics were identified as a stiff system, with the flow dynamics in the order of 3-4 times faster than the rotor dynamics. The backward difference method allowed for convergence at larger time steps.

Improvements to DIGTEM led to the development of DEAN; A Program for Engine Dynamic Analysis [66]. DEAN made use of a highly modularised aerothermodynamic model of gas turbine systems. This vastly improved the flexibility of the software, allowing for various types of engine configurations and components to be analysed individually or as a whole. DEAN also included a better user interface, which was easier to use and allowed for direct interaction between the user and model, even during a simulation. DEAN, however, still operated on the fundamental principles of DIGTEM. Nonetheless, an improvement in fidelity was introduced with each new version of software. Some of those advanced features are discussed in the section that follows.

4.2.3 Enhanced Fidelity

The methods presented thus far, have demonstrated reasonable accuracy in predicting the performance of engine non-linear dynamics. However, throughout the literature, various authors have incorporated different ways in which to improve the fidelity of such simulations. This subsection serves to discuss some of those improvements found in literature.

Variable Specific Heats

Usually, engine models assume that gasses are calorically perfect. Although a reasonable approximation, this is not true, as specific heat is a function of both temperature and pressure. However during engine simulations, the exit pressures and temperatures for a component are unknown, and their calculation is in-fact dependent on the ratio of specific heats. Therefore,

an iterative procedure is required to determine the gas properties at the outlet of each component. This is normally avoided to curb the penalties in computational speed incurred through the iterative calculations required per component, per time step.

Rahman [67] included variable specific heats in a model of a high-bypass, three-spool turbofan engine. This model was claimed to be a combination of the iterative and inter-component volume methods, and was referred to as a "hybrid" method. The advantage of this hybrid method was, apparently, to take advantage of the accuracy of the iterative method and computational efficiency of the inter-component volume method. This author is of the opinion that Rhaman has misunderstood the definition of the Iterative and ICV Methods. The iterative method is actually a lower fidelity model as it does not incorporate any form of volume dynamics. Furthermore, the inter-component volume method is in fact slower because of the "stiff" problem created by the combination of fast volume dynamics and slower rotor dynamics. In addition, neither methods are defined by the manner to which individual component aero-thermodynamics are calculated. Instead, the Iterative Method uses an iterative procedure for component matching within a single time step, whereas the ICV Method allows this to occur dynamically (which is more physically representative) through time variant accumulation of mass. Therefore, by definition, Rhaman's model is not a hybrid of the two, but is instead the ICV Method, with the inclusion of improved fidelity; by accounting for variable specific heats. Although not stated explicitly in the literature reviewed thus far, it is assumed that, due to its obvious nature, variable specific heats have been incorporated in models prior to the one developed by Rahman.

Nonetheless, iterative methods for calculating specific heats are presented by Rhaman [67] and Flack [33]. Flack presents two methods of calculating the variable specific heats, based on the assumption that specific heats very with temperature only:

$$\Delta h = \int_{T_i}^{T_e} c_p \, \mathrm{d}T \tag{4.18}$$

Equation 4.18 may be approximated by using either the end states, Equation 4.19, or average values, Equation 4.20. In accordance with the Midpoint integration rule, it is evident that Equation 4.20 will yield a better approximation. Therefore, Flack recommends converging toward the components average thermodynamic properties, when iteratively determining specific heats. The error associated with this approximation is depicted in Figure 4.12.

$$\Delta h = (c_p T)_e - (c_p T)_i \tag{4.19}$$

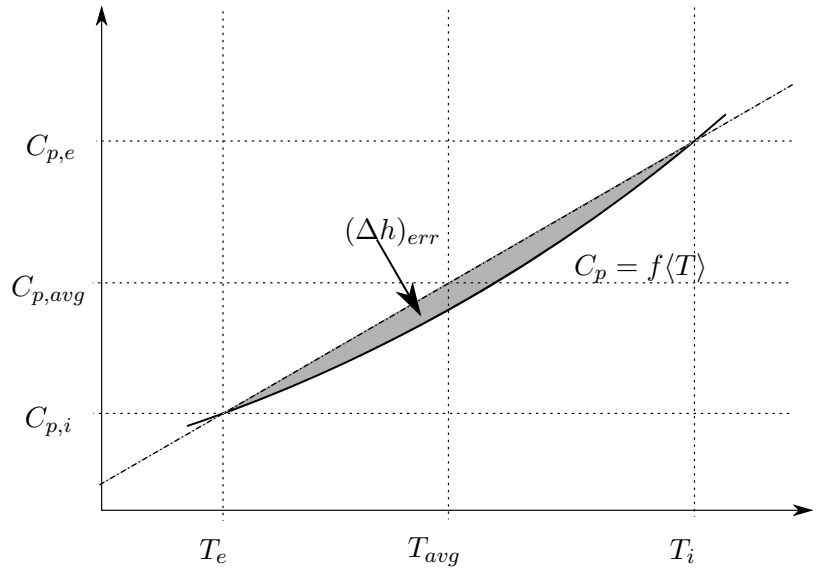


Figure 4.12: Error in change of enthalpy due to approximation of constant, average specific heat values

$$\Delta h = \bar{c}_p \Delta T \tag{4.20}$$

Rahman on the other hand, uses a different method, which calculates enthalpy change across the component as a function of both pressure and temperature. This method first calculates the isentropic operation of the component, to which the isentropic efficiency is later applied. However, as per Flack's argument, this method is based on end state values and does not take advantage of the Mid-point integration rule. Flack's method can be easily expanded to take into account the effects of pressure, as well as temperature, on specific heat values. Specific heats, as a function of pressure and temperature, may be determined from piecewise curve fits (valid over different temperature ranges) or alternatively as tabulated data. Methods for accurately calculating the specific heat of a gas is summarised by Heywood [68].

2D Representation of Fan Characteristics

The fidelity of turbofan simulations may be improved by the representation of the fan's flow characteristics. Initially, the fan was represented using a single compressor map, as is the case with the other compressors in the engine. Such maps average the flow conditions entering and exiting a compressor, resulting in 0-dimensional approximation of uniform flow. This may have been suitable for engines with small bypass ratios, however larger fans exhibit stronger radial profiles, which must be accounted for. Visser [69]

found that a single-map fan representation may yield inaccuracies of up to 10%. An improvement on the single-map representation was made by simply introducing two separate maps; one representing the core and bypass flows, respectively. These maps are constructed, in a similar manner to the single-map representation, using profile averaging techniques, and provides a more accurate representation of the flow characteristics entering the bypass duct and engine core. Twin-map representations are usually accompanied by a curve relating BPR and engine speed (determined from steady-state engine tests) or alternatively calculated. These variations in BPR effect simulation accuracies significantly. Visser [69] was able to reduce inaccuracies from 10% to less than 2.5%, using twin map representation with calculated variations in BPR.

Alternative methods, proposed by Yin [70, 71], make use of a what has been termed as a *fan-radial profile performance map*. For a given operating condition, this method calculates the radial profile of flow properties from a single characteristic map. The bypass ratio is determined iteratively to match the flow demands of the downstream components. A comparison of all three representations was conducted by Curnock *et al.* [72]. The first method, using a single map representation, tended to over-predict exit pressures and therefore, overall engine performance. Steady-state simulations indicated very little difference between the results of the other two representations. However, Curnock *et al.* predict that the radial-profile map will allow for more accurate predictions of BPR during engine transients. Further investigations of 2-dimensional fan representation have been investigated, and are discussed in detail in References [73–76]. However, the twin map method is utilised in GSP software (which has been used to validate the model developed in this work), and therefore this work will follow the method described by Visser [69].

Volume Dynamics

The aero-thermodynamic coupling of engine components involves highly interdependent relationships, which occur in one continuous control volume; from intake to exhaust nozzle (or two control volumes in the case of a bypass engine). In most instances, the operation of the components are simplified to a zero-dimensional description, using steady-state component maps. This simplification, in effect, divides the overall control volume into sub-volumes which exist between components. These sub-volumes serve to "communicate" operating conditions between adjacent components. This "communication" volume may be modelled according to the laws of mass, momentum and energy conservation. This has already been discussed, to a degree, so far. Fawke and Saravanamuttoo [51, 59] were the first to introduce mass accumulation and its effect on rates in pressure changes in the ICVs. Sellers and Daniele [61] extended this fidelity by incorporating the effects of

energy storage in DYNGEN and later included momentum conservation in DIGTEM [64]. These conservation laws are discussed here in further detail, with reference to its application in subsequent literature.

First, it is necessary to analyse the control volume from first principles, to which the reader is referred to Anderson [15], and we begin with the conservation of mass, and the continuity equation. Considering a fixed control volume, the continuity equation may be given in the form of a partial differential equation; Equation 4.21

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{4.21}$$

Equation 4.21 holds true for unsteady, viscous and compressible flow in three dimensions, where the velocity vector is given in terms of its Cartesianal components:

$$\boldsymbol{V} = u\hat{\boldsymbol{i}} + v\hat{\boldsymbol{j}} + w\hat{\boldsymbol{k}} \tag{4.22}$$

Equation 4.21 may be simplified to Equation 4.23, assuming one-dimensional, uniform flow in the axial direction of the volume; i.e. $u, v, {}^{\partial(\rho \boldsymbol{V})}/_{\partial y}, {}^{\partial(\rho \boldsymbol{V})}/_{\partial z} = 0$.

$$\frac{\partial \rho}{\partial t} = -\frac{\partial (\rho u)}{\partial x} \tag{4.23}$$

The term in the partial differential, on the right-hand-side of 4.23, may be substituted by $\rho u = {}^{\dot{m}}/_{A}$ to yield Equation 4.24.

$$\frac{\partial \rho}{\partial t} = -\frac{\partial \left(\frac{\dot{m}}{A}\right)}{\partial x} \tag{4.24}$$

Obeying the quotient rule, Equation 4.24 may be expanded to give Equation 4.25.

$$\frac{\partial \rho}{\partial t} = -\left[\frac{A\frac{\partial \dot{m}}{\partial x} - \dot{m}\frac{\partial A}{\partial x}}{A^2}\right] \tag{4.25}$$

A further simplification may be to assume that both the area and mass flow rates vary linearly between the inlet and outlet of the control volume. In addition, it is assumed that the density throughout the volume is uniform. Therefore, linear gradients and averaged values of mass flow rate and area, may be substituted into Equation 4.25:

$$\frac{\partial \rho}{\partial t} = -\frac{\left[\frac{(A_i+A_e)(\dot{m}_e-\dot{m}_i)-(\dot{m}_i+\dot{m}_e)(A_e-A_i)}{2L}\right]}{\left[\frac{A_i+A_e}{2}\right]^2} \tag{4.26}$$

Where L is the length of the volume in the axial direction. Rearranging and simplifying Equation 4.26, yields Equation 4.27.

$$\frac{\partial \rho}{\partial t} = \frac{\left[\frac{A_e \dot{m}_i - A_i \dot{m}_e}{L}\right]}{\left[\frac{A_i + A_e}{2}\right]^2} \tag{4.27}$$

However, often the inlet and exit areas to the volume are unknown. Therefore a final assumption may be made to assume a constant area throughout the volume. This final assumption gives Equation 4.28, which gives the rate of change of the volumes density as a function of its volume and difference between inlet and outlet flow rates.

$$\frac{\partial \rho}{\partial t} = \frac{\dot{m}_i - \dot{m}_e}{V} \tag{4.28}$$

Next the conservation of momentum in the volume is considered. The general momentum equation, only in the axial direction, is represented as a partial differential equation given by Equation 4.29 (obtained from Anderson [15] as before).

$$\frac{\partial (\rho u)}{\partial t} + \nabla \cdot (\rho u \boldsymbol{V}) = -\frac{\partial p}{\partial x} + \rho f_x + (F_x)_{viscous} \tag{4.29}$$

Where f_x represents the body forces, per unit mass, exerted on the fluid inside the volume in the x (axial) direction. And $(F_x)_{viscous}$, represents the viscous forces in the x (axial) direction. Assuming inviscid flow with uniform properties in the radial direction, Equation 4.29 becomes:

$$\frac{\partial (\rho u)}{\partial t} = -\frac{\partial \left(\rho u^2\right)}{\partial x} - \frac{\partial p}{\partial x} + \rho f_x \tag{4.30}$$

It is noted that $\rho u = \frac{\dot{m}}{A}$ and may be substituted into Equation 4.30, to yield:

$$\frac{\partial \dot{m}}{\partial t} = -\frac{\partial}{\partial x}\left[\dot{m} u + pA\right] + \rho A f_x \tag{4.31}$$

Once again, it is assumed that all gradients vary linearly over the length of the volume and are applied to average values in the volume. In addition, the body forces on the fluid body (exerted by "distant" forces such as gravity) are neglected and assumed zero. Therefore, Equation 4.31 is simplified and rearranged to give Equation 4.32.

$$\frac{\partial \dot{m}}{\partial t} = \frac{(\dot{m} u + pA)_i - (\dot{m} u + pA)_e}{L} \tag{4.32}$$

The conservation of energy may now be analysed. It's general equation, in partial differential form, is obtained once again from Anderson [15] (ignoring viscous forces):

$$\frac{\partial}{\partial t}\left[\rho\left(e+\frac{|\boldsymbol{V}|^2}{2}\right)\right]+\nabla\cdot\left[\rho\left(e+\frac{|\boldsymbol{V}|^2}{2}\right)\boldsymbol{V}\right]=\rho\dot{q}-\nabla\cdot(p\boldsymbol{V})+\rho\left(\boldsymbol{f}\cdot\boldsymbol{V}\right) \tag{4.33}$$

Equation 4.33 is simplified by assuming inviscid, adiabatic, one-dimensional (in the axial direction) flow. In addition, body forces are neglected as before. Given these assumptions, Equation 4.33 becomes:

$$\frac{\partial}{\partial t}\left[\rho\left(e+\frac{u^2}{2}\right)\right]+\frac{\partial}{\partial x}\left[\rho\left(e+\frac{u^2}{2}\right)u\right]=-\frac{\partial}{\partial x}\left(pu\right) \tag{4.34}$$

However the total enthalpy is given by:

$$h_t=h+\frac{u^2}{2}=e+p\upsilon+\frac{u^2}{2} \tag{4.35}$$

Which is substituted into Equation 4.34 to yield Equation 4.36.

$$\frac{\partial}{\partial t}\left(\rho h_t-p\right)=-\frac{\partial}{\partial x}\left(\rho h_t u\right) \tag{4.36}$$

Expanding both sides of Equation 4.36, and rearranging, gives the rate of change of enthalpy of the volume, as indicated in Equation 4.37.

$$\rho\frac{\partial h_t}{\partial t}=-\left[h_t\frac{\partial\rho}{\partial t}-\frac{\partial p}{\partial t}+h_t\frac{\partial\left(\rho u\right)}{\partial x}+\rho u\frac{\partial h_t}{\partial x}\right] \tag{4.37}$$

Then, by substituting $\rho u=\frac{\dot{m}}{A}$ into Equation 4.37, and assuming a constant cross-sectional area, one may obtain Equation 4.38.

$$\rho\frac{\partial h_t}{\partial t}=-\left[h_t\frac{\partial\rho}{\partial t}-\frac{\partial p}{\partial t}+\frac{h_t}{A}\frac{\partial\dot{m}}{\partial x}+\frac{\dot{m}}{A}\frac{\partial h_t}{\partial x}\right] \tag{4.38}$$

As was done with the continuity and momentum equations, it is assumed that the gradient between end states is linear and that the time derivatives are calculated for average states of the volume. Furthermore, the gas may be assumed to be calorically perfect. These assumptions result in a simplification of Equation 4.38 to Equation 4.39.

$$\frac{\partial T_t}{\partial t}=-\frac{\left[h_{t,\mathrm{avg}}\frac{\partial\rho}{\partial t}-\frac{\partial p}{\partial t}+\frac{\Delta(h_t\dot{m})}{V}\right]}{\rho c_p} \tag{4.39}$$

Finally, the gas inside the volume is assumed to behave in accordance with the ideal gas law:

$$p=\rho RT_t \tag{4.40}$$

Which, when derived with respect to time, becomes:

$$\frac{\partial p}{\partial t} = R\left[T_t\frac{\partial \rho}{\partial t} + \rho\frac{\partial T_t}{\partial t}\right] \tag{4.41}$$

Therefore, the thermodynamic state of the volume may be determined by integrating a set of partial differential equations described by Equations 4.28, 4.32, 4.39 and 4.41. In doing so, the average values, axially across the volume, of pressure, density, temperature and mass flow rate are known. These volume thermodynamic states are then used to define the operating point of adjacent components.

Similar expressions may be found in Alexiou and Mathioudakis [77] as well as in the documentation which describes DYNGEN [61], DIGTEM [64] and DEAN [66]. Alexiou and Mathioudakis do not describe exactly how they use these expressions, however the documentation on DYNGEN, DIGTEM and DEAN do give some insight as to their usage. The discussion below is the authors intepretation as to how these equations should be used, although it must be mentioned that some insight was gained from the relevant references already mentioned.

There is the dilemma of choosing which variables should be used to define the operation of the adjacent components. Remembering that the compressor and turbine maps are given for values of pressure ratio, corrected inlet mass flow rate, isentropic efficiency and corrected operational speed; there are two options as to how the operating point may be defined. The operational speed may be used along with either pressure ratio or corrected mass flow rate, i.e. :

$$(\dot{m}_C;\eta) = f\left\langle N_C; PR\right\rangle \text{ OR } (PR;\eta) = f\left\langle N_C;\dot{m}_C\right\rangle \tag{4.42}$$

Using either of the expressions of Equation 4.42, will yield atleast one contradiction between the thermodynamic states calculated from the set of partial differential equations and those obtained from component characteristic maps. However, a choice may be justified by revisiting the assumptions made during the simplification of the continuity, momentum and energy equations. It was assumed that pressure and density are uniform throughout the volume and that the momentum equation yields the average axial flow rate through the volume. Therefore, the first option in Equation 4.42 may be used, by assuming that the conservation of momentum is maintained by the automatic adjustment of the axial flow velocities across the length of the volume i.e. $u(x)$. Hence, only Equations 4.28, 4.39 and 4.41 are required to deal with the dynamics in the inter-component volumes; which account for mass and energy storage.

The analysis may be simplified further, by assuming that the volume's inlet and exit temperatures are equal (not forgetting that a calorically perfect gas has already been assumed). Therefore, the energy equation, Equation

4.39, may be simplified through substitutions of Equations 4.28 and 4.41, to yield Equation 4.43.

$$\frac{\partial T}{\partial t} = \frac{RT}{\rho c_v V}\left(\dot{m}_i - \dot{m}_e\right) \tag{4.43}$$

The rate of change of pressure may then be obtained through substituting Equations 4.28 and 4.43 into Equation 4.41:

$$\frac{\partial p}{\partial t} = \frac{\gamma RT}{V}\left(\dot{m}_i - \dot{m}_e\right) \tag{4.44}$$

Equation 4.44 is also valid for the rate of change in actual pressure and not just the specific pressure. The actual pressure may be expressed in terms of a partial differential equation, Equation 4.45, from which Equation 4.46 is obtained by substituting $m = \rho V$.

$$\frac{\partial P}{\partial t} = \frac{R}{V}\left(m\frac{\partial T}{\partial t} + T\frac{\partial m}{\partial t}\right) \tag{4.45}$$

$$\frac{\partial P}{\partial t} = R\left(\rho\frac{\partial T}{\partial t} + T\frac{\partial \rho}{\partial t}\right) \tag{4.46}$$

Therefore, substituting Equations 4.28 and 4.43 into Equation 4.46, gives Equation 4.47 which has the same right-hand-side as Equation 4.44.

$$\frac{\partial P}{\partial t} = \frac{\gamma RT}{V}\left(\dot{m}_i - \dot{m}_e\right) \tag{4.47}$$

Equation 4.47 is very similar to that derived by Saravanamuttoo and Fawke [51], Equation 4.13, except for the γ term in the numerator of the first term. In addition, equating Equations 4.45 and 4.47, it is found that:

$$m\frac{\partial T}{\partial t} = \frac{\gamma - 1}{\gamma}\frac{\partial P}{\partial t} \tag{4.48}$$

This means that the $m\dot{T}$ term in Equation 4.13 accounts for $\frac{\gamma-1}{\gamma}\%$ of the pressure rate of change. For $\gamma = 1.4$, this term equates to, approximately, a 29% contribution to the rate of change in pressure. This is close to the percentage, 20%, quoted by Fawke *et al.* [60]. Therefore, Equation 4.47 may be used to predict the change in pressure ratios across components by approximating the mass and energy storage in the inter-component volumes. This is how volume dynamics will be incorporated into the engine model developed in this work.

Heat Soakage and Variable Tip Clearances

Heat soakage is the transfer of heat between the working fluid and the engine components. The effects of heat soakage become prominent during slam accelerations/decelerations, meaning that dynamic simulations, based on steady-state data, are no longer accurate [39]. Such effects are also related the the blade tip clearances, due to varying expansion/contraction rates, and changes in combustion efficiency.

Varying blade tip clearances occur during engine transients due to differential expansion rates of blades, seals and engine casings [39].

It is common practice to assume that the operating temperatures of the component hardware matches that of the working gas. However, it is clear that this is not the case and that a heat flux exists between the two mediums. In fact, secondary flows are bled from compressors to cool turbines to allow for higher turbine inlet temperatures which increases engine performance. Heat soakage effects the thermodynamic cycle both directly (change in gas temperature) and indirectly (expansion rates of hardware has geometrical effect). It is therefore important to investigate the impact that heat transfer has on the accuracy of performance predictions.

Alexiou and Mathioudakis [77] and Kurzke [78] discuss a simplified method of including heat transfer effects within a transient simulations. In this method, the geometry of hardware is simplified so that blades are represented as plates and ducts as cones, all of uniform thickness. In addition, all hardware is assumed to distribute temperatures uniformly (i.e. infinite thermal conductivity). The heat flux, due to convection, is then determined from Newton's Cooling Law as given by Equation 4.49.

$$\dot{q} = \varphi A \Delta T(t) \tag{4.49}$$

In Equation 4.49 $\dot{q}$, φ and A denote the rate of heat transfer, heat transfer coefficient and component area of convection, respectively. $\Delta T(t)$ represents the difference in temperature between the component material and the working gas, both of which are time variant. Equation 4.50 is determined by expressing Equation 4.49 in terms of the components heat capacity.

$$\frac{dT}{dt} = -\frac{\varphi A}{C} \Delta T(t) \tag{4.50}$$

Assuming that the gas temperature is relatively constant compared to that of the hardware, Equation 4.50 is integrated to yield Equation 4.51. This represents a first order lag with time constant $-\frac{\varphi A}{C}$. Note that the heat capacity may be substituted for specific heat; $C = c_p m$.

$$\Delta T(t) = \Delta T(t) e^{-\frac{\varphi A}{C} t} \tag{4.51}$$

Kurzke explains how one might obtain the varying heat transfer coefficient as a function of Reynolds Number Index. Nonetheless, Kurzke also

advises that this calculation method may not yield accurate results because of the assumptions made, and therefore the heat transfer coefficient, or even the time constant itself, should be altered via trial-and-error to obtain more realistic results.

Alexiou and Mathioudakis present an explicit equation for determining the heat transfer coefficient; Equation 4.52. In this equation c, Re, K and L symbolise a constant, Reynolds number, thermal conductivity and reference length respectively. The Reynolds number may be obtained from Equation 4.53, where A represents the cross-sectional area.

$$\varphi = 0.0021 c Re^{0.8} \frac{K}{L} \tag{4.52}$$

$$Re = \frac{\dot{m}_g L}{\mu A} \tag{4.53}$$

Heat transfer between the gas and components alters the temperature of the gas, and therefore directly effects the thermodynamic performance of the components and engine.

During a transient, the casings and rotary blades expand/contract at different rates. This is because the casing and ducts are subjected to thermal loading only, whereas blade elongation results from a combination in centrifugal and thermal stresses. Blade elongation due to centrifugal forces occurs more rapidly than the thermal expansion of the casings. This difference in expansion rates in-turn varies the clearances between the blades and their casing. In so doing, the flow characteristics of the compressor/turbine is effectively modified to yield a different efficiency and, of particular importance, surge margin [78].

Another influence of heat transfer, is that of turbine blade cooling and compressor bleeds. The percentage of compressor bleed cannot be assumed to be constant as it is dependent on the turbine inlet and blade temperatures. In addition, when a compressor is run at higher pressure ratio's, its exit temperature is higher and therefore a greater amount of bleed is required to keep the turbine temperature below its operational temperature limit. This is illustrated by the parametric study by Kurzke [79], which is used to demonstrate the cycle optimization of a typical helicopter gas turbine engine.

The thermal effects mentioned above change parameters, which the engines control system uses to dictate fuel flow metering rates. Therefore, inaccuracies may be introduced by excluding such effects.

Nielsen et al. [80], investigated thermal effects in greater detail, considering both blade tip and seal clearances. Finite element methods are used to predict changes in geometry of casings, shafts, disks, blades and seals. Changes in blade tip clearance influence flow leakage in compressors and turbines, which in-turn influences the efficiency of the rotary components.

Furthermore, the capacity of the compressors and turbines are reduced and increased, respectively, for increases in blade tip clearances. These effects are used to modify steady-state performance maps by changing surge and choke margins as well as efficiency contours. Modified component maps are included in the normal transient engine model to improve fidelity. Variations in bleed and fuel flow rates, due to seal geometric changes, are also accounted for. Linearised state space models of the finite element analysis on components, are used for improved computational efficiency. The inclusion of thermal effects by Nielsen *et al.* was validated against engine test data. A cold stabilization manoeuvre (cold engine accelerated quickly to take-off power) was used as the validation case.

Bringhenti and Barbosa [81] present a turbine specific analysis of blade tip clearance effects, to which the reader is referred for a more detailed description of tip clearance effects.

Nevertheless, one can gather from the above that the inclusion of heat soak age within an engine model is no simple feat, and therefore is omitted from this work.

Inlet Distortion Effects

Inlet distortion occurs when the flow enters the engine non-uniformly. This may occur either circumferentially, radially or as a combination of both. Distortion may be viewed as a variation in pressure ratio in non-stream wise directions. A simplified technique for evaluating the effects of distortion is what is referred to as the *Parallel Compressor Model*. The parallel compressor model is briefly outlined by reference [82]. Each area of varied pressure is modelled by a separate compressor map. The operating point on these individual maps are defined such that the compressor exit pressure is uniform. Therefore, a distorted area of lower inlet pressure correlates to a high pressure ratio operating point on the same speed line as the map describing the high inlet pressure distortion. This results in a cascaded effect onto the following component. The cascade efffect is now no longer represented as a pressure distortion but rather as a temperature distortion, due to the pressure rise resulting from varied pressure ratios. Inlet distortion may effect the operating stability of the compressor, particularly when surge is encountered by the region with low inlet pressure.

Inlet distortion becomes an important factor particularly for extreme aircraft manoeuvres, as in the case of a fighter jet. However, commercial airliners are less likely to encounter inlet distortion that significantly effects its engine's performance. Generally such conditions occur at take-off or landing, and not at cruise, which is the flight regime of interest to this work. However, it must be noted that when flying into the wake of another aircraft, the inlet distortion may become a likely factor influencing engine performance and, in a more detailed analysis, should be considered. The

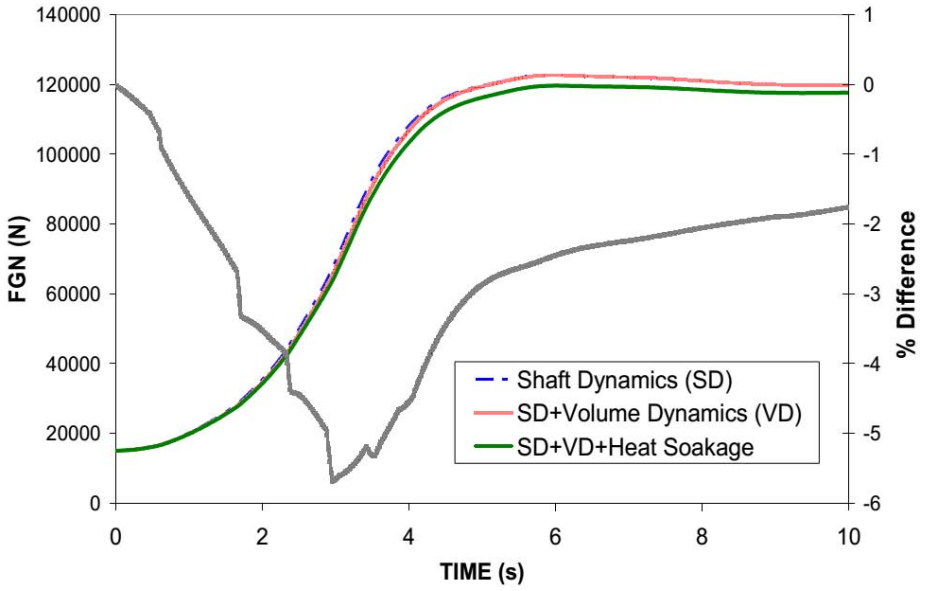


Figure 4.13: A comparison of including the effects of volume dynamics and heat soakage. Reprinted from Alexiou and Mathioudakis [77]

reader is also referred to the section on intakes in Appendix E.2, for a more in-depth discussion on inlet distortion.

Discussion

Variable specific heats and twin map fan models, are simple enough to include within an engine model for the purposes of this work. The next question is whether it is necessary to account for volume dynamics and heat soakage effects?

The effects of volume dynamics and heat soakage on engine performance was examined by Alexiou and Mathioudakis [77]. A step response of a turbofan engine revealed slightly different responses for three cases. The first case considers rotor dynamics only, the second includes volume dynamics and finally heat transfer is incorporated in the third case. The results are repeated here in Figure 4.13.

This figure illustrates that the inclusion of volume dynamics marginally slows the response time of the engine. This is understandable as the volume dynamics are substantially faster than the rotor dynamics. The heat soakage effects result in a slight reduction in thrust, which makes sense as the thermal efficiency of the engine decreases with heat losses to the components. Most importantly, it must be noted that deviations from the rotor dynamic predictions never exceed errors of 6%. This raises the question as to whether it is indeed necessary to include these higher fidelity effects?

Alexiou and Mathioudakis [77] include a frequency response analysis, which partially answers this question. The thrust response of the engine was examined for a frequency range in sinusoidal fuel flow. These were performed for an simulation including only rotor dynamics and another incorporating volume dynamics. For the engine model only modelling rotor dynamics, a fairly consistent change in gain across the frequency range was observed. However, once volume dynamics were include, a change in this trend was noted for high frequencies ($> 20Hz$) of fuel flow. The higher order effects of the engines non-linear dynamics are therefore only significant at high frequencies. This seems reasonable, as the volume dynamics are low in inertia and quick to respond to changes in fuel flow. Whereas, rotor dynamics respond a lot slower in comparison. Therefore at lower frequencies the non-linearities of the engine is dominated by the rotor dynamics.

This frequency analysis may lead to the conclusion that it depends on the operational conditions of the engine as to whether volume dynamics should be included or not. The effects of heat soakage on frequency response was, however, not considered. It would be interesting to perform a similar frequency response analysis to determine heat transfer affects. Further more, the effects of variable tip clearance would surely alter the response significantly.

Therefore, based on the above and the context of this work, it is important to include volume dynamics within the model. This is because the thrust demand in response to formation dynamics and turbulence, may result in dynamic fuel flow commands of a high enough frequency that yield significant differences in performance predictions between models including and excluding volume dynamics.

The improvements in accuracy in incorporating heat soakage effects, however, does not warrant the complexities and effort of its inclusion, at least at this initial stage of the work. For similar reasons the effects of inlet distortion have been ignored in this initial investigation. However it is recommended that subsequent high-fidelity investigations into engine performance during formation flight, seriously consider the effects of inlet distortion, particularly concerning the onset of compressor surge. Therefore, of the possible ways to improve the engine model's fidelity (as mentioned above), variable specific heats, twin-map fan representation and volume dynamics have be included.

4.2.4 High Fidelity Simulations

The simulation methods presented thus far have been aimed at predicting realistic response-time characteristics of gas turbine transients. The more sophisticated models, those used to predict non-linear dynamics, have all made use of component steady-state characteristic maps. The fidelity of such models was initially considered to be sufficient due to the close agreement

achieved between engine test data and simulated results. However, as the demand for improved engine efficiency and performance increased, so did the complexities in engine configurations, component designs and control systems. As a result, higher fidelity simulations are required to aid the development of new engines.

GETRAN

A code that was successful in drastically improving fidelity was GETRAN, which was developed in the early 1990's [83]. The code made use of generic modules that were modelled mathematically using the continuity, energy and momentum equations. A one-dimensional approximation of each module was developed according to an assigned category of operation. The first category, indicative of nozzles and intakes, accounted for modules that did not transfer energy either mechanically or thermally, thereby only including conversions between kinetic and potential energy. The second category included thermal energy transfers, as is the case with combustion chambers. The third category, representative of rotary components, accounted for both mechanical and thermal energy transfers. Modelling of ancillary components, such as valves, sensors and control systems, was also included.

Special attention was paid to the modelling of the compressors and turbines, using a modular approach to model the stages of the compressor/-turbine separately. These were modelled on a row-by-row, commonly know as stage-stacking, basis, which made use of stage steady-state performance characteristics. In this way, variable geometry effects could be analysed and intelligently manipulated by control systems during transient simulations. An example of this would be active surge control, which varies stator blade angles to avoid surge. Furthermore, depending if cooling was utilized, heat transfer effects could be included by using a diabatic process approach. Adiabatic models were kept as an option for turbo-machines without cooling.

The inter-component volume type approach was extended to incorporate the modelling of plenums. A plenum is essentially a volume, wherein the flow dynamics between two adjacent modules are coupled. It incorporates not only the volumes located between modules, but also that which is considered to be part of the module. For instance, a compressor is surrounded by a plenum, which couples the compressor stages. A rewording might name the plenum control volume as an inter-module volume instead of an inter-component volume.

GETRAN was also capable of changing between levels of fidelity, which is defined by the level of complexity used to model the components. The first level makes use of component steady-state performance maps with no flow dynamic coupling (i.e. mass/energy storage). This is akin to the iterative method mentioned earlier. The second level, also using component steady-state maps, incorporates some flow dynamics by allowing for mass

and energy storage in volumes between components. This is akin to the inter-component volume method referred to earlier. The third level replaces steady-state component maps with a one-dimensional approximation of the continuity, energy and momentum equations. This approximation assumes that components operate under adiabatic conditions. Finally, in the fourth level, components are assumed to operate diabatically. This is effectively the inclusion of heat transfer between component blades and working fluids. The level of fidelity achieved by GETRAN was validated using experimental data.

Numerical Propulsion System Simulation (NPSS)

During the early 1990's, it was noted that propulsion development was completely reliant on large scale experimental testing, which resulted in huge costs. Furthermore, experimental and simulation limitations prevented a complete analysis of component interactions, which could be of extreme importance. It was recognised that computational technology was improving at a rapid rate and would soon be able to meet the demands of fidelity, thereby replacing much of the testing required. A framework, referred to as the Numerical Propulsion System Simulation, was initiated to facilitate the developments in complete and high fidelity propulsion simulations [84]. This framework was to be built as a collaborative effort between NASA, industry and Universities.

Claus et al. [85] recognised that high fidelity analysis is heavily dependent on a multidisciplinary collaboration, including; aerodynamics, structural analysis, materials, heat transfer and control systems. Four areas were highlighted as key to the success of the framework to be developed; data interfacing, code modularity via object oriented languages, integration of multi-fidelity models and high performance computing using distributed an parallel processing.

Drumond *et al.* [86], in 1992, motivated the benefit of approaching gas turbine simulation from an object-oriented point of view for the development of the NPSS project. At the time, most simulations of the time were programmed using procedural based languages, predominantly FORTRAN. The procedural based languages of the time suffered the dilemma of compromising either accuracy or computational speed. Of the obstacles mentioned, three were highlighted in particular as those which an object-oriented language could overcome.

The first obstacle was the modelling of component performance characteristics from first principles. Procedural languages had resorted to utilizing steady-state performance maps, rather than evaluating performance from first principles, to allow for acceptable simulation times.

The second obstacle was that regarding the dynamic flow and thermodynamic coupling between adjacent components. Due to the fast time con-

stants of these dynamics, procedural numeric solvers needed to use small time steps to allow for convergence and stability of solutions, thereby reducing computational speed.

The third obstacle highlighted by Drummond *et al.* [86], involved the interdisciplinary coupling between various technologies. The object-orientated approach introduced modularity that allowed for the use of both distributed and parallel processing. This means that higher fidelity simulations may be run without a compromise in computational speed. In addition, it was envisaged that object oriented languages would allow for the inclusion of multi-fidelity models within the same simulation. Furthermore, the hierarchical structure improves program maintainability, flexibility and code re-usability. The simulation program DIGTEM [64], discussed toward the end of the previous section, was developed as a prototype to demonstrate some of the capabilities of object-oriented gas turbine simulations.

The development of the NPSS project was to involve two aspects. The first was to model each component of the engine at different levels of fidelity. The second aspect was to integrate the component models, which may be run on parallel processors or separate systems and in different programmes or languages. Homer and Schlichting [87, 88] introduced an initial "simulation executive", the purpose of which was to provide a coupling between component models and to centralise the control and interface of the entire engine simulation.

A prototype code, STEPP [89], was written to investigate the effectiveness of object-oriented language for high fidelity simulations. The aim of this prototype was to create a one-dimensional model that could support dynamic multi-fidelity component calculations, in a procedure referred to as "zooming". The model was component-based, with objects representing those of actual engine components, thus motivating a control volume approach to component simulation. Furthermore, there was to be no limitation on the number of components included, and the architecture was to allow for flexibility in modelling various engine configurations. The objects included were those of components, ports and connectors. Components are able to receive and output information via ports, and the connectors are used to transmit this information between components. The connectors are "setup" by the user to define the engine's configuration and operation. This paper on STEPP discusses different methods for accommodating mass and energy storage between adjacent components, appropriate to the object-oriented structure. STEPP initially only contained models of inlets, ducts, nozzles, compressors, shafts and electric motors to simulate compressor rig tests. This was deemed as sufficient for the purpose of prototyping the framework required. Although no example of its application was given, the prototype was claimed to be successful in-terms of a development phase in the NPSS project. STEPP was not intended to directly introduce any improvements in fidelity, but rather was designed as a framework to allow for

the integration of multidisciplinary high-fidelity component models.

One of the primary focuses of NPSS was to implement zooming. Zooming is the ability of a simulation to dynamically shift between component models of different fidelity. This shift may be controlled automatically by the simulation executive, according to the transient state, or alternatively manually by a user.

Reed et al. [90] give a description of how zooming may be applied to the fan of a turbofan simulator, using a prototype simulator. Here, a one-dimensional aerothermodynamic engine model is used as a low fidelity simulator that calculates the operation of the overall engine. The fan's operation, in this example, may be calculated from steady-state performance maps (low-fidelity) or alternatively a three-dimensional Navier Stokes numerical solution (high-fidelity). The one-dimensional engine model calculates the inlet flow conditions to the fan, which are single average values of total temperature, pressure and Mach number. The engine model also determines the exit total pressure, thus defining the pressure ratio across the fan. These values are extrapolated to define a three-dimensional boundary condition for the Navier Stokes equations. The high-fidelity zoomed code then determines the exit conditions of the fan, which are averaged to give a uniform output from the fan. If the pressure ratio given by the averaged outlet pressure is equivalent to that dictated by the one-dimensional engine model, then the other properties of mass flow and temperature given by the three-dimensional analysis may be used for the remainder of the overall engine simulation. However if this is not the case, which is often likely, then the three-dimensional fan simulation must be run iteratively to find a solution that matches the specified pressure ratio. Unfortunately, this iterative procedure proved to be unstable and oscillatory, thereby lacking suitable convergence.

To curb this issue, the high-fidelity fan model was used to calculate the speed line at the specified rotor speed, thereby reducing the number of interpolations required, as well as subsequent interpolation errors (the error during speed interpolation is discussed in detail by Fawke and Saravanamuttoo [51]). This is because performance maps are bivariate and need to be interpolated for both spool speed and pressure ratio. Parallel computing is used to determine the numerous operating points making up the speed line, thus reducing the computational time required. Nevertheless, it was envisaged that the convergence and stability issues encountered by the the iterative use of the three-dimensional fan calculations would soon be overcome.

Reed and Afjeh [91] created a framework of their own, called Onyx, to accomodate their developments in high fidelity simulations. This framework was written in JAVA and superseded the STEPP framework [89] referred to previously. The continued development of Onyx by these authors is described in references [92, 93].

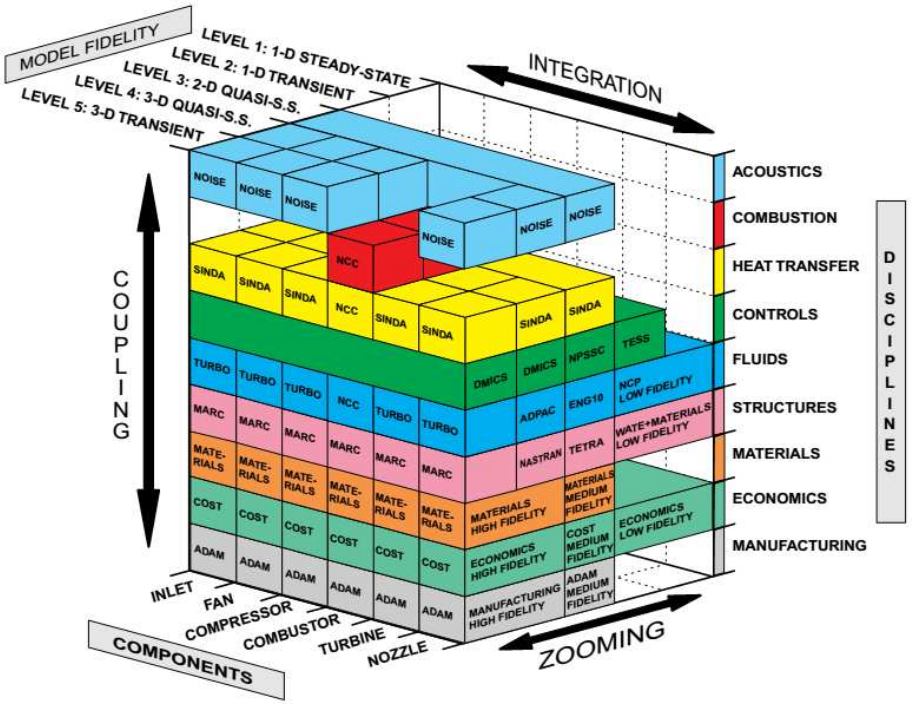


Figure 4.14: The topology of the NPSS system, relating the multidisciplinary and multi-fidelity structure and coupling of the desired simulation programme. Reprinted from Lytle [94]

An overview of the progress achieved during the 1990's an a projection of future work moving into the 21^{st} century is documented by Lytie [94, 95]. Lytle summarises the the desired topology of the NPSS project, which is repeated here in Figure 4.14. A roadmap relating the development of this topology is also repeated here in Figure 4.15. These figures have been included as they best summarise the the NPSS project.

A further overview, by Follen [96], details the use of NPSS and its applicability to investigating space propulsion, rocketry, ground power generation and the development of weapon systems.

The NPSS project has been developed extensively since its conception, and the number of works involved in its evolution are vast and specialised. One could dedicate an entire literature review, devoted solely to the coverage of the NPSS project. However, the level of specialisation contained in such works is beyond that required of the engine model for the purposes of this work. Therefore, the reader is referred to the overviews provided by Claus *et al.* [97, 98] for a comprehensive look some of the more recent advancements achieved through the NPSS program.

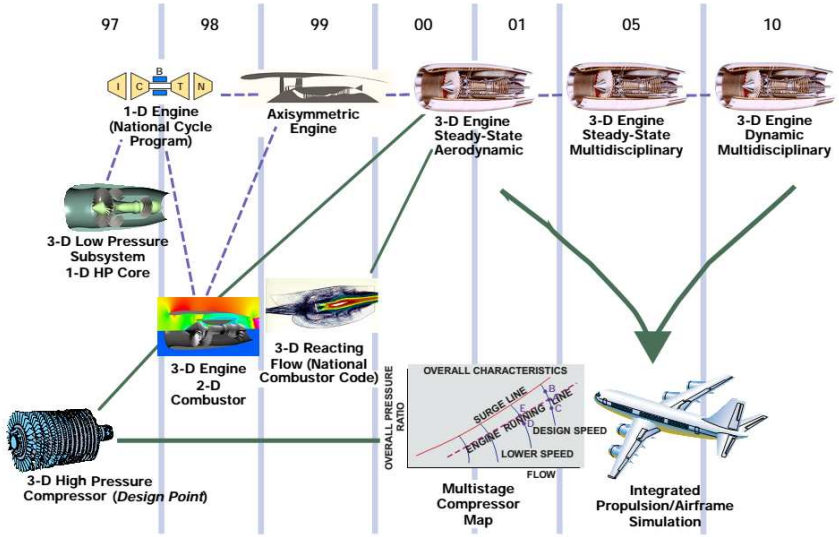


Figure 4.15: The intended roadmap devised for the development of the NPSS project in 1999. Reprinted from Lytle [94]

4.2.5 Available Software Packages

Many of the simulation codes and programs mentioned in the literature review, were developed by universities and industry toward their respective research aims. Therefore, such codes are not readily available for use by external parties. This section serves to discuss some of the software packages that are currently available to the public commercially.

MAPSS and C-MAPSS

The Modular Aero-Propulsion System Simulation (MAPSS and C-MAPSS, C for Commercial), is a component-based simulation model developed by NASA in the MATLAB/SIMULINK environment. This package is a basic fidelity simulation of a typical military and commercial turbofan engine, which makes use of typical compressor and turbine performance maps. The intention of this software was to provide a platform for control system analysis and engine diagnostics. It is freely available on request, but may not be released to non U.S. citizens. Therefore the author of this work was unable to utilize this software package. The reader is referred to References [99–103], for more details on this software.

GasTurb

An overview of GasTurb is given in reference [79], and is summarised here. GasTurb is a commercially available software package that attempts to find a

balance between over-simplified gas turbine models and those incorporating unnecessary fidelity. An emphasis has been placed on user-friendliness via an easy-to-understand graphical user-interface. This emphasis is founded on the idea that the success of a simulation is dependent on its use. The GasTurb package is targeted at private, educational and industrial use. GasTurb may be used for parametric studies as well as off-design steady-state and transient performance analysis of different engine configurations; turbojet, prop, fan (mixed or unmixed flows).

A case study on the cycle optimization of a helicopter gas turbine engine was presented to demonstrate the effectiveness of GasTurb in performing such parametric studies. This study examined, more closely, the effect of turbine cooling when determining optimum compressor ratios.

GasTurb allows for the calculation of off-design steady-state operating conditions using component performance maps. A basic description of the component matching technique, with reference to a single-spool turbojet engine, is given. Component matching techniques are all similar, however Kurzke makes use of a different state variable. This state variable, is the value of the auxiliary coordinate β, which is used in representing bivariate component maps digitally, and aids the interpolation of the digitized data. This method has been used in the engine model developed in this work, and is discussed in more detail in Appendix D. In terms of component matching, the β parameter is used in place of either pressure ratio or mass flow rate for defining the operating points of the turbo-machine components.

GasTurb also allows for the analysis of inlet distortion on engine performance, through the addition of a parallel compressor model. Distortion has already been discussed, in the previous section.

The transient calculations of the GasTurb model is an extension of the off-design steady-state matching procedure. Engine accelerations are based on power imbalance and rotor dynamics (i.e. iterative method). The effects of volume dynamics, heat transfer and variable tip clearance had been omitted to avoid complication of the model. However, these effects are to be included in future versions.

GasTurb has included a PID controller, which is used to manage fuel flow rates in response to changes in the power level angle (PLA). The PLA is representative of the thrust output of the engine. The fuel flow rate is controlled according to maximum engine pressures, temperatures and acceleration/deceleration rates.

One of GasTurb's most notable contributions to gas turbine modelling, is its accompanying software Smooth C [104] and Smooth T [104], which is used for handling compressor and turbine performance maps, respectively. A unique method (as already mentioned) was devised to curb issues of collinearity, encountered during performance map interpolation. This methodology is outlined by Kurzke [82], the software developer of GasTurb. Other software packages, such as GSP and PROOSIS, also make use of the

performance map handling methodology developed by Kurzke.

A more comprehensive description of GasTurb may be found in the user manual [105], which includes details, such as the mathematical component models, numerical solvers and details of steady-state and transient methodologies.

PROOSIS

PROOSIS first began as a component library to the simulation environment Ecosimpro^{TM}, which was used to perform steady-state and transient simulations of a twin-spool turbofan engine [77]. A summary of the description given in reference [77] is presented here.

Ecosimpro^{TM} is an generic, object oriented simulation environment written in C++. Models may be coded in a high level language or alternatively constructed graphically by dragging and dropping components from libraries onto the graphical user interface. Ecosimpro^{TM} has built in solvers to deal with non-linear differential equations. It is also cable of connecting with other simulation environments such as MATLAB/SIMULINK.

The object oriented architecture is comprised of ports and components, which make use of inheritance, aggregation and polymorphism to avoid code duplication, improve re-usability and maintainability. Ports are defined according to the variables required for communication between components. This includes gas composition and thermodynamic properties as well as power demand and supply between compressor-turbine pairs.

The turbomachine components are modelled using steady-state performance maps and mathematical descriptions of their associated thermodynamics. Included are the effects of Reynolds number on component performance. The fan is modelled using a twin-set of maps, and is assumed sufficiently large so as to ignore the effects of Reynolds number. The characteristic maps are handled using those methods described by Kurzke [82], and may be interpolated using either linear, cubic or spline interpolation schemes.

The combustion is modelled according to the air-fuel ratio and the heating value of the fuel. The combustion efficiency is assumed constant, but pressure losses are included. These losses are a function of Mach number and a pressure loss coefficient. In addition, the component library includes ducts and mixers (for mixed flow turbofans), which are modelled using basic 1D flow theory.

Engine transients calculations include rotor and volume dynamics along with heat soakage effects. Volume dynamics calculated inside inter-component control volumes, based on conservation of mass, energy and continuity. Heat soakage effects are determined using Newton's Cooling Law. Both volume and hgeat soakage effects may be switched on or off according to the level of fidelity required.

The turbofan engine model, created in the initial development stage of PROOSIS, was validated against an industry accepted model [106]. The error in steady-state predictions did not exceed 0.5% and good correlations were demonstrated for transient compressor map trajectories.

Alexiou and Mathioudakis [107], discuss the various features of the gas turbine library developed in reference [77], along with various applications of the simulation tool. These applications include the development of a new engine model from existing components, including a new component (cooled turbine is introduced) and integrating the engine model with other applications outside of Ecosimpro^{TM}.

The engine modelling library, developed in Ecosimpro^{TM}, was moved to its own specialised, standalone platform: PROOSIS. This initial PROOSIS prototype is detailed in reference [108], which presents an overview of the product along with its improved capabilities. Particular attention is paid to the addition of component zooming and distributed processing. A compressor stage-stage stacking code is used to demonstrate multi-fidelity decoupled, semi-coupled and fully-coupled zooming capabilities. Zooming was performed remotely using the Web Services executive to control distributed processing, which communicated between a centralised 0-D engine model and a remote 1-D stage-stacking compressor model. Details of advantages and disadvantages are given along with recommendations for the development of PROOSIS. PROOSIS was first released commercially in 2008 with improved versions released annually.

PROOSIS would have provided the perfect tool for the engine modelling requirements of this work, due to its integratibility with other environments such as MATLAB/SIMULINK (used in modelling aircraft dynamics). Unfortunately only a free/student version of Ecosimpro^{TM} is available, which does not include components from the PROOSIS library, and no funding was available to purchase PROOSIS.

GSP

The Gas Turbine Simulation Program (GSP) [69,109–111], developed at the Nationaal Lucht- en Ruimtevaartlaboratorium (NLR - National Aerospace Laboratory) in the Netherlands, is a commercially available software package for simulating gas turbine operation.

An overview of GSP's initial development is provided by reference [69], and is summarised here. The motivation for GSP's development, in 1986 at the Delft Technical University, was to improve upon the instabilities and user-interface of the DYNGEN code, which has been discussed previously. Therefore, the mathematical models used in GSP are inherently the same as those found in DYNGEN. GSP is a component based model, making use of steady-state component characteristic maps, which are used to analyse and predict steady-state and transient performance of various engine con-

figurations. Like DYNGEN, GSP is a one-dimensional model that assumes uniform flow characteristics throughout the engine. A set of non-linear, differential equations, with time-based derivatives, are formulated based on power, mass and energy imbalances, from which engine transients are predicted.

Along with the improvement of stability and user-interface, GSP also introduced secondary effects; compressor bleeds, turbine cooling, heat transfer and internal volume effects. However, one of the more significant improvements was the manner in-which the fan was modelled [69]. This has already been discussed within this literature review, and is revisited in Appendix E.4. The GSP model was verified through its application of various analyses on different types of engines [69]. The results indicated that inaccuracies of less than 1% and 5% were achievable for steady-state and transient simulations respectively.

Later, GSP was rewritten in an object-oriented language, Borland Delphi, allowing for better architecture, re-usability and maintainability. This was accompanied by the introduction of a high-fidelity combustion model for better predicting gas turbine emissions [109]. Further developments in GSP are included in the overview in reference [110]. This overview gives explanation to the object-oriented architecture used, component models and gas models. The functionality of GSP is demonstrated through examples of its application. This included a parametric study of high-bypass turbofan engine take-off performance in response to variations in ambient temperature and compressor bleeds. The effects of including volume dynamics and heat soakage was demonstrated by the application of altering a turboshaft engine to include heat recuperation. Further devlopment of GSP, with particular attention to adaptive modeling, is presented in reference [111].

GSP is a generic engine modelling tool and does not include the multi-fidelity features of the NPSS and PROOSIS packages. It does however include multidisciplinary aspects, including control system modelling. Furthermore, users are able to custom-build engine components for the inclusion in simulations.

The author had been granted a GSP student licence. Unfortunately, the student version of GSP is in a standalone environment and can only be accessed via its GUI. Therefore, it was not possible to directly include it within the aircraft models developed in Simulink. As a result, there was no existing engine model available to include within the aircraft and formation model. Therefore, the author chose to replicate some of the GSP software in the MATLAB/Simulink environment. Subsequently, the engine model developed was validated against GSP.

4.3 Summary and Discussion

The first section, discussed the overall workings of the gas turbine jet engine. This included understanding the different types of gas turbine engines available and how they operate in both steady-state and transient conditions. This was necessary in order to understand the objectives for an engine model.

Finally, a comprehensive review of gas turbine simulation, from its beginning in history to present, was conducted. The purpose of this section, was to gather as much information about the manners in which gas turbine engines have been modelled. In addition, it was also important to investigate all possible codes and software packages that may have been available for use in this work.

From this literature review, the various simulation techniques were revealed, along with their applicability and relevance to certain objectives. This aided the decision as to how, if necessary (i.e. if there was no commercial or other software package available), an engine model required for this work should be developed and what features it should include. It was discovered that the engine model should be based on a thermodynamic component model approach. To that end, suitable accuracy could be achieved by implementing the Inter-Component Volume method (thereby incorporating volume dynamics), based on steady-state characteristics of the components. In addition, it was found that the fidelity of the model could be enhanced by accounting for variable specific heats within component models as well as using a twin-map to model the performance of the fan. It was concluded that any improvements beyond these additions, would be too complex for practical inclusion within a model developed specifically for this work.

Nonetheless, a search was done to see if there was any existing software, available to the author, that would provide equal or better fidelity. It was found that many packages do exist, however only one software package was available to the author due to budget constraints. This package was GSP, which was obtained on a student licence. However, it was found that GSP did not provide the compatibility required to be incorporated within the aircraft models of this work. As a result, it was concluded that an engine model needed to be developed from scratch, but that GSP could provide a useful reference.

Chapter 5

Turbofan Engine Model

As already eluded to in previous chapters, aircraft will constantly have to readjust there throttle settings in order to maintain their station within a formation submersed in atmospheric turbulence. It is intuitive to expect that dynamic throttling will reduce engine operational efficiency, thereby increasing the cumulative fuel required to achieve the same amount of average thrust were the aircraft flying in steady-state conditions. This has obvious implications on the possible benefits that may be achieved through flying aircraft in formation, and therefore an accurate prediction of the engine's performance, during dynamic operation, is required. This chapter describes the engine model which has been developed to accurately predict the fuel consumption due to dynamic throttling in formation flight.

5.1 Model Requirements

The engine model developed for the purposes of this work, is required to accurately predict the transient performance (at varying altitudes and airspeeds) of a typical high-bypass turbofan engine, which is used to power the commercial airliners of today. The model should also be able to predict, to a reasonable extent, the higher order effects which are inherent in its non-linear behaviour. To this end, the model should be thermodynamically based and account for rotor and volume dynamics.

In addition, the engine model developed must be capable of integrating easily with an aircraft model developed in the Simulink environment, and therefore should be programmed in the MATLAB/Simulink environment. In saying so, the model should be self contained and interface simply with the aircraft model via inlet (fuel flow, altitude and airspeed) and outlet (thrust) ports. Hence, its contents should be contained within a single Simulink block, that may be easily incorporated within other Simulink Models.

Furthermore, it is required that the engine model should allow for a certain amount of adaptability, in that it may be used to simulate engines

of different types and or configurations. To this end, it should include a library of engine components which may be easily assembled together to form different configurations. Finally, the model must be user friendly and allow for easy manipulation by user without compromising the integrity of critical calculations.

5.2 Model Description

A thermodynamic, component-based approach was taken to model the engine. The Inter-Component Volume Method, was selected as it allowed for the inclusion of volume dynamics in addition to rotary dynamics. Volume dynamics are necessary to improve the accuracy of predicting higher-order non-linearities in the engine transients, particularly in response to oscillatory fuel inputs (as has already been discussed in the literature review). Such oscillatory fuel inputs may be a result of the dynamic throttling required by an aircraft to maintain it station within a formation.

Each component of the engine was modelled mathematically, based on its thermodynamic operation. The majority of the thermodynamic components found in an engine, have been extensively described and validated in Appendix E. These include; intake, fan, compressors, combustion, turbines and nozzles. The performance of the rotary components (i.e. compressors and turbines) were modelled based on steady-state characteristic maps, which has been discussed extensively in Appendix D.

What remains, is to assemble these components in a meaningful way, such to represent a turbofan engine. To this end, there are additional components to be developed, which are necessary for coupling the afore mentioned components, mechanically and thermodynamically. These additional components are the Inter-Component Volumes and Spools, which are the fulcrum of engine dynamics.

5.2.1 Inter-Component Volumes (ICV's)

The inter-component volume is used to thermodynamically couple adjacent engine components. These volumes model the effect of flow accumulation, which may occur in-between components who's mass flow rates are mismatched. The theory of Inter-Component Volumes has already been discussed extensively in Section 4.2.3, and therefore is not included here to avoid repetition. Therefore, the volume components are used to represent Equation 4.47, which is repeated here for convenience:

$$\frac{\partial P}{\partial t} = \frac{\gamma RT}{V}\left(\dot{m}_i - \dot{m}_e\right) \tag{5.1}$$

In addition, the volume component allows for a pressure loss term. The purpose of this is to allow the user to include any pressure losses that may

occur between components. An example would be the inclusion of pressure stacking losses, prior to entry to the exhaust nozzle.

Equation 5.1 and the pressure loss term, are self contained within a Simulink Block, and have no accompanying MATLAB code. The Simulink Blocks of the volume component are depicted in Figure 5.1. Figure 5.1a depicts the highest level system, used for interfacing with other engine components. It takes in the flow conditions exiting the preceding component, as well as the mass flow rate of the component downstream of itself. It outputs the the thermodynamic changes to the fluid, as well as the bus signal of its preceding component. This is necessary, as the component adjacently downstream of the volume often requires information about the component upstream of the ICV. An example, is the passing of information about the gas' composition from combustion to high pressure turbine, to low pressure turbine, and finally on to the nozzle. This is necessary, because the ICV does not 'store' information on the gas composition in its own bus signal.

The ICV component outputs a total pressure which is independent of the pressure loss term. This pressure is used by preceding components (e.g. compressors and turbines) in determining their pressure ratios, which are in-turn required to define their off-design operation from steady-state characteristic maps.

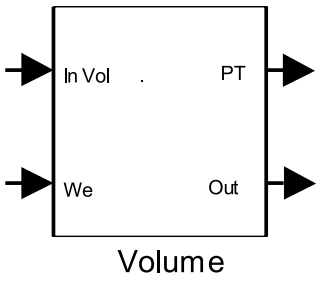
Figure 5.1b depicts the user-interface level of the block. Here, the user may define the size of the volume as well as any pressure losses that may occur. This pressure loss term is to be specified as a pressure ratio across the volume. Also at this level, the user is required to specify a name to which the component will record and export its data.

Finally, Figure 5.1c depicts the subsystem which performs the calculations specified in this section. This subsystem is quite simple and therefore does not warrant any further explanation. However, it is noted that the user is required to supply an initial value for the pressure inside the volume, corresponding to the initial operating point of a transient simulation. Transient simulation initialisation is discussed further on.

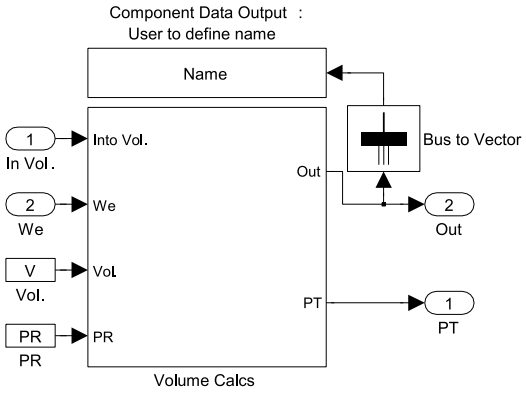
Because the volume is a 'coupling component', it is not applicable to validate it as an isolated component. Its validity is dependent on the validity of the engine model as a whole.

5.2.2 Spools

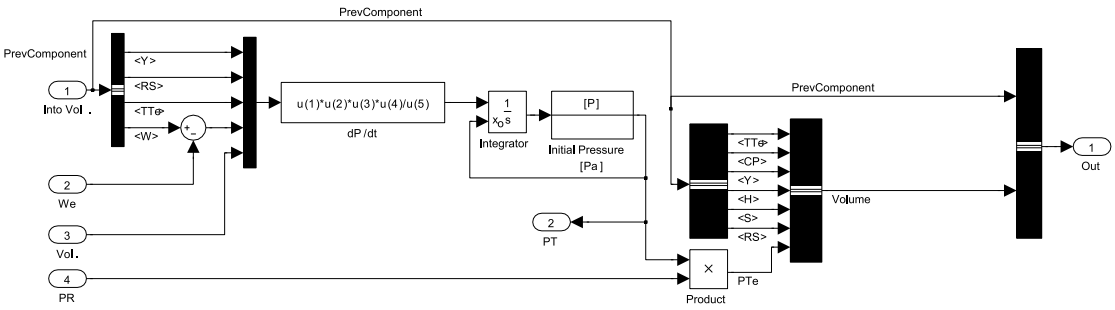
Compressors and turbines are physically coupled to one another via spools. The turbines supply torque (power) to the spool, and compressors extract power from the spool, according to Equation E.11. There are, however, mechanical losses in transferring the power from the turbines to compressors. Therefore, an efficiency term is incorporated to account for such losses and to yield the actual power available to the compressors attached to that spool, Equation 5.2.



(a) High-level volume block



(b) User-interface level



(c) Volume calculations subsystem

Figure 5.1: Inter-Component Volume model developed in Simulink

$$\mathbb{P}_c = \eta_m \mathbb{P}_t \tag{5.2}$$

However, during transient operation the power supplied by the turbine does not necessarily equal the power extracted by the compressor. It is this mismatch which causes the spool to either accelerate or decelerate, according to Equation 5.3.

$$\dot{N} = \frac{\eta_m \mathbb{T}_t - \mathbb{T}_t}{I} \tag{5.3}$$

Equation 5.3 has been developed into a Simulink block, and is depicted in Figure 5.2. Figure 5.2a, illustrates the highest level of the spool system used for interfacing with other components. It takes in the sum of the torques of the compressors and turbines coupled to it, and it returns the running speed after integrating Equation 5.3.

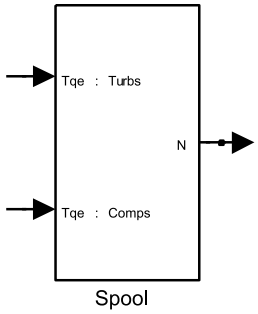
Figure 5.2b shows the details of the subsystem of Figure 5.2a, and is the level for interfacing with the user. Here the user is required to specify the moment of inertia of the spool (this includes the components attached to it) and mechanical efficiency at which it transfers power from the turbines to the compressors.

Finally, Figure 5.2c depicts the details within the subsystem 'Spool Calcs'. Equation 5.3 is contained within this subsystem and includes an integrator. As with the ICVs, the user is required to specify an initial speed at which the spool operates, at the start of a transient simulation (this is discussed in more detail later).

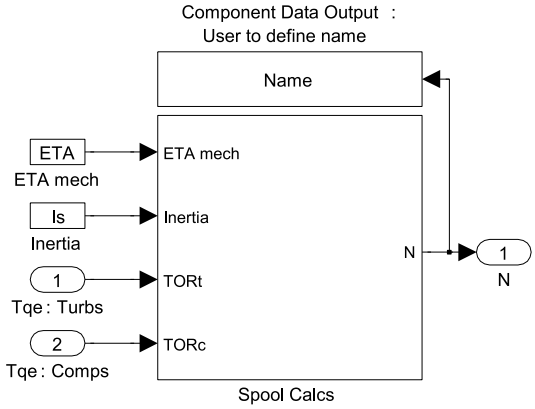
5.2.3 Turbofan Engine Component Assembly

Now that all the components have been developed into Simulink blocks, they may be combined to form a complete engine model. To begin with, Figure 5.3 is a schematic of the component assembly of a typical twin-spool, high-bypass, turbofan engine model. This schematic serves to provide the reader with a reference for the Simulink block assembly that follows, thereby aiding the visualisation of how the actual Simulink model represents an engine. Each component in the schematic has been labelled, apart from the volumes which have been filled with a light grey colour. The numbers on the schematic are there to represent/name the different stations along the gas path/s passing through the engine.

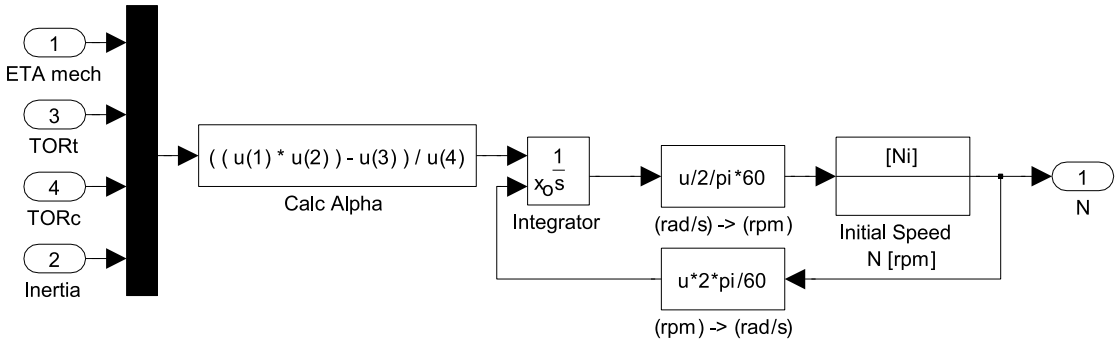
Now, with reference to Figure 5.3, the same configuration is represented as an assembly of the Simulink component blocks developed in this work, Figure 5.4. Figures 5.3 and 5.4 have been colour coded, such that they correlate with one another. In addition, the tags in Figure 5.4 (used for signal routing) are also colour coded according to the colour of their source block. This allows one to easily follow the flow of signals between blocks.



(a) High-level spool block



(b) User-interface level



(c) Spool calculations subsystem

Figure 5.2: Spool model developed in Simulink

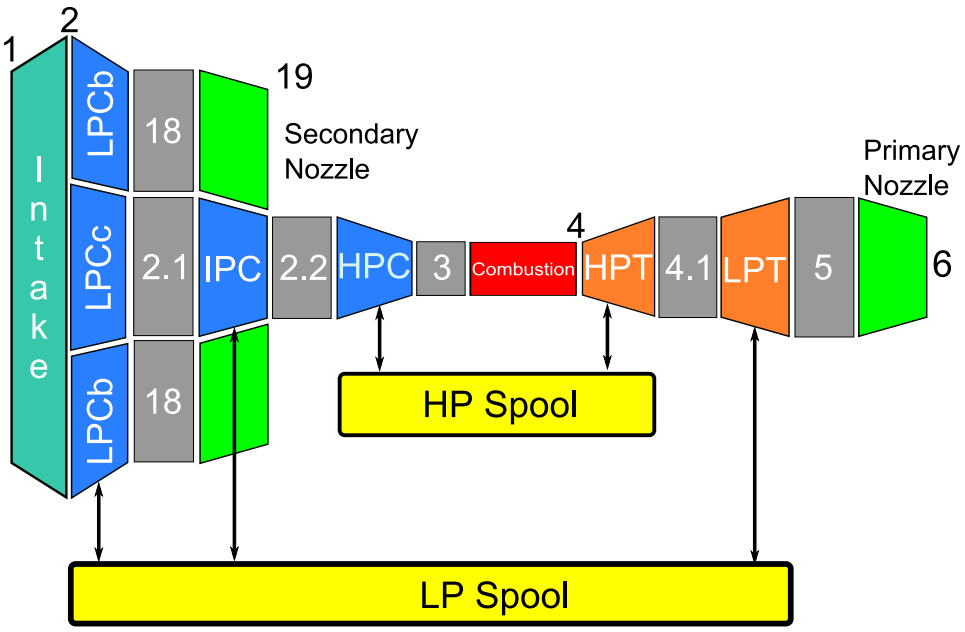


Figure 5.3: A schematic of an assembly of gas path components representing a typical twin-spool, high-bypass, turbofan engine model

Figure 5.4, is actually the contents of a Simulink block, Figure 5.5, which is used to interface with aircraft models.

Figures 5.3 and 5.4 will be used as a reference when explaining the engine model and how the components interact with one another. This explanation follows the gas path, beginning at the intake and moving downstream to the nozzle components.

Flight Conditions and Intake

Naturally, the first port-of-call is to establish the atmospheric conditions, in which the engine is submersed. This is determined by the flight conditions (FC) block, which takes in the inputs of airspeed and altitude, to determine the stagnation temperatures and pressures as well as other corresponding thermodynamic parameters. These atmospheric conditions are then ingested by the engine via the intake (passed between blocks using a bus signal), which serves to increase the static enthalpy of the flow reaching the fan's inlet. The flight conditions block also outputs the local atmospheric static pressure via a tag, which is used to determine the thrust at each nozzle. In addition the intake calculates the ram drag experienced, which is passed,via a tag, to the portion of the model which calculates the net thrust produced by the engine. A zoomed illustration of the flight conditions intake assembly is depicted by Figure 5.6. As can be seen from this figure, the mass flow rate that passes through the intake is determined by the amount 'demanded' by

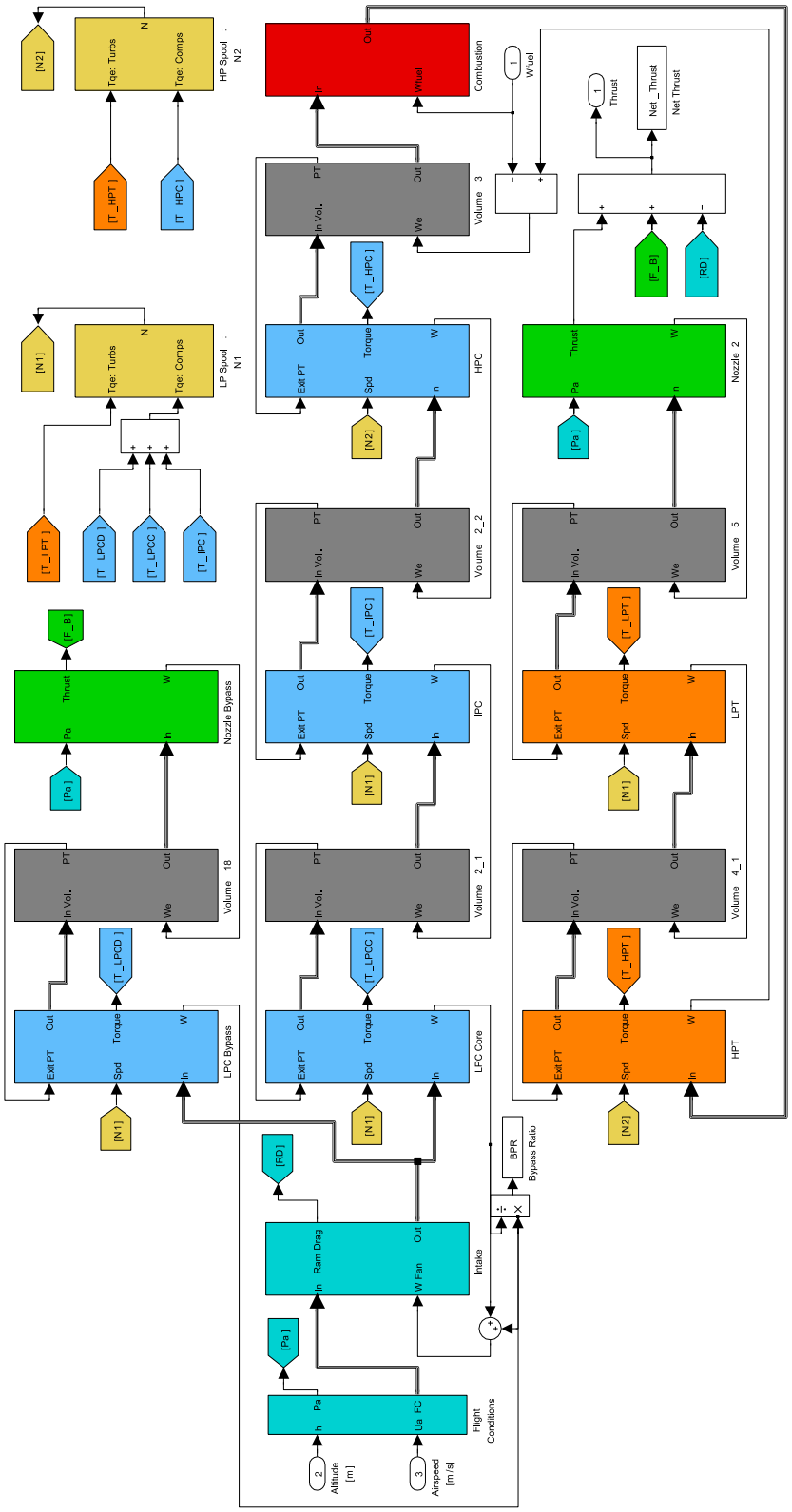


Figure 5.4: Twin-Spool High-Bypass Turbofan Engine Model Assembly in Simulink

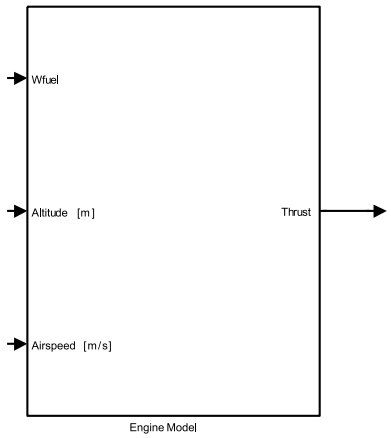


Figure 5.5: High-level engine model simulation block, for interfacing with aircraft models

the fan. In saying so, it is assumed that the intake's capture ratio adjusts instantaneously to accommodate the flow requirements of the fan.

The Fan Component

The theory associated with the fan component has already been discussed in Appendix E.4, but did not include a Simulink block assembly. This is because the components developed in Appendix E, are intended to be general, thereby allowing them to be combined in different ways to simulate different engine configurations according to different modelling methods (inter-component volume or iterative methods for example).

The fan component assembly found in Figure 5.4, has been isolated and illustrated in Figure 5.7. This figure has included the intake as well as the two ICV's found downstream of each of the parallel compressors making up the fan. This is to show how the fan interacts, necessarily, with its adjacent components.

In this figure, it is shown how the fan is comprised of two compressor components. Each of these components contains a different compressor map, which accounts for the fan's prominent radial flow profile. This is done by averaging the portion of that profile which each compressor represents.

Uniform flow is assumed to reach the inlet of the fan, which is evident by the equal flow properties passed to the inlet of each compressor from the intake. It can also be seen, that each compressor takes in its exit pressure from a corresponding downstream ICV. The ICV determines this pressure based on the accumulation of mass resulting from a mismatch in flow entering (from the fan's compressor) and exiting (from the bypass nozzle or intermediate pressure compressor) the ICV.

The operation of the fan's compressors are therefore defined by its op-

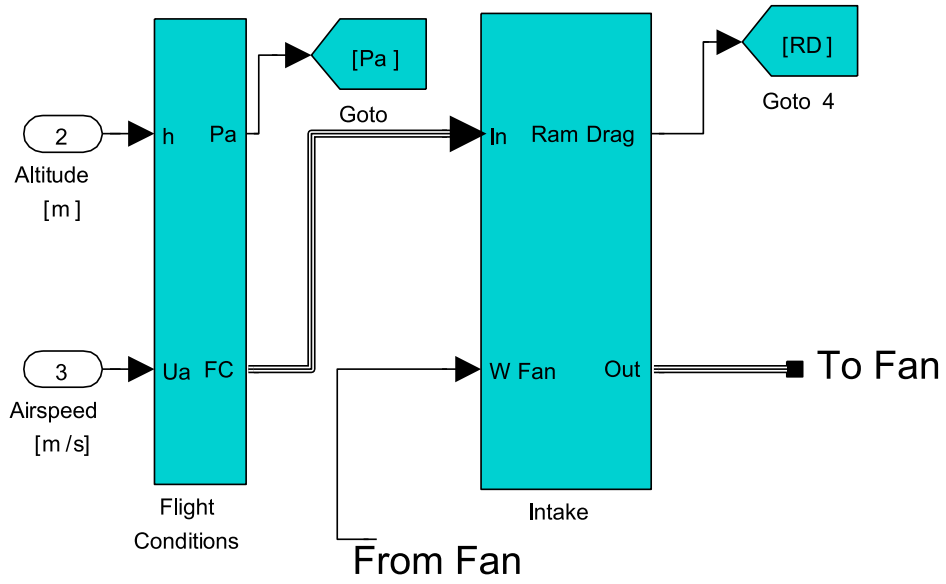


Figure 5.6: Flight conditions and intake block assembly, zoomed in from Figure 5.4

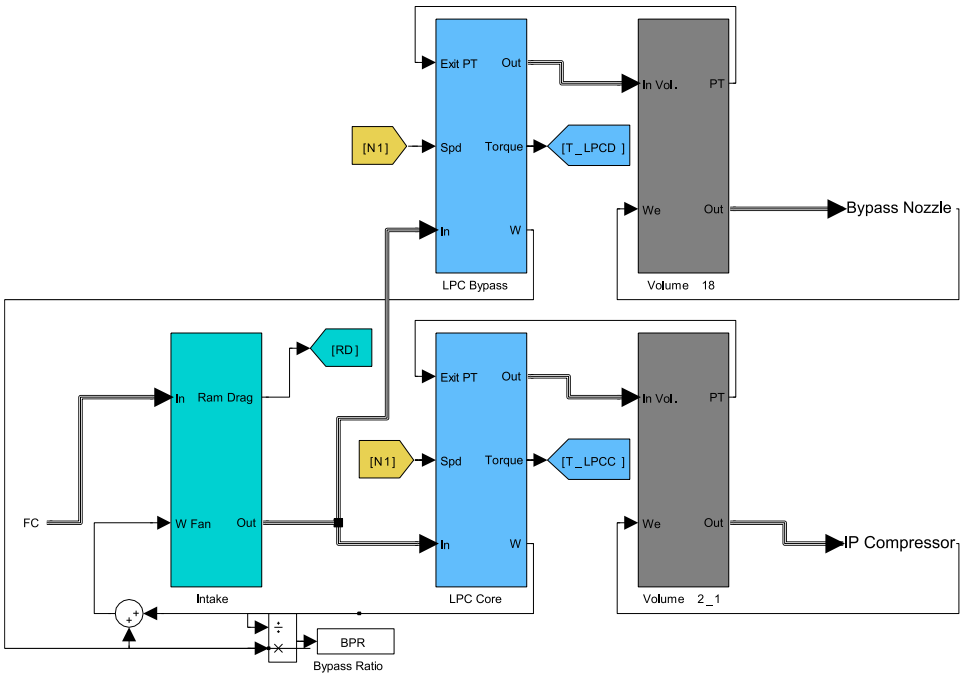


Figure 5.7: Fan Simulink Block Assembly, zoomed in from Figure 5.4

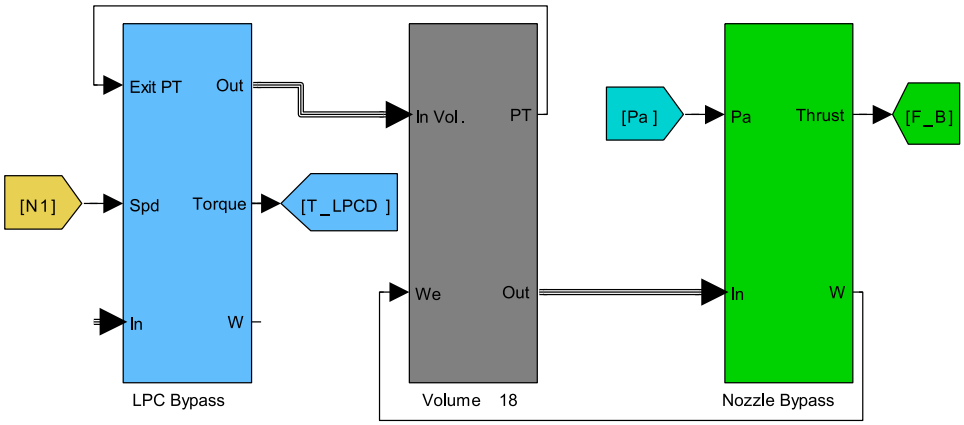


Figure 5.8: Bypass Gas Path Simulink Block Assembly, zoomed in from Figure 5.4

erating speed, pressure ratio and inlet temperatures and pressures. The resulting exit flow properties are then passed on to the subsequent volume, and the torque to the corresponding spool component. Finally, the mas flow rates of each compressor are combined to give the intake's mass flow rate, and are also used to determine the engine's bypass ratio at that specific operating point.

Bypass Flow Path

When considering Figures 5.3 and 5.4, it must be remembered that there are two separate gas flow paths, which are split from the original gas flow path in the fan. Figure 5.8 illustrates the bypass gas path and its components.

This assembly includes the fan's bypass compressor and the bypass nozzle, which are separated by an ICV. The rate of change in pressure within the ICV is determined by the accumulation of mass, which is a result of a mismatch in the flow rate permitted by the nozzle and that given by the compressor. Often, this stacking of pressure before the nozzle may result in some pressure losses which are accounted for in the ICV block. The pressure derivative in-turn effects the operation of the compressor and nozzle resulting in interdependent dynamics. Finally, the inlet total pressure and temperature (along with corresponding gas thermodynamic properties) are used to determine the thrust generated at that nozzle.

Core Compressors

Next, we consider the primary gas path, which passes through the engine's core. To begin with, the compressors downstream of the intake are considered. These include the low (core of fan), intermediate and high pressure

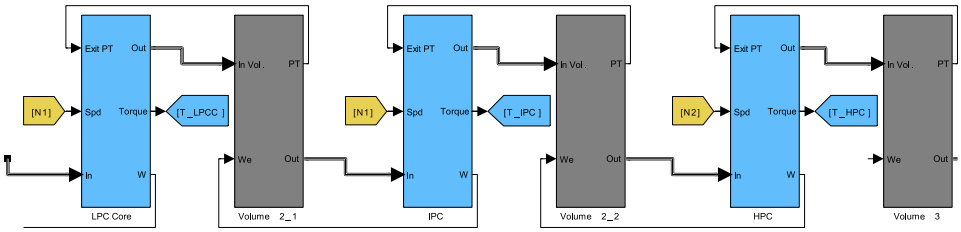


Figure 5.9: Core Compressors Simulink Block Assembly, zoomed in from Figure 5.4

compressors, which are each separated by an ICV, as illustrated by Figure 5.9.

There is an obvious pattern in the flow of information between these components. Information flows in both upstream and downstream directions. Each component receives information on its inlet conditions from its preceding component (i.e downstream flow of information). Whereas, some of the information, regarding exit conditions, is received from subsequent components (i.e. upstream flow of information). Information on exit pressure (and therefore pressure ratio) and exit flow rate is passed upstream to the compressors and ICV's respectively. This pattern clearly indicates the interdependence between adjacent components. The pressure ratio of each compressor, is dependent on the pressures within the volumes surrounding each it. And the pressure within each volume, is dependent on the flow rates to and from its surrounding compressors. It is this interdependency (and similarly with the turbines), which dictates the volume/flow dynamics through the engine.

In addition, the compressors and spools are also interdependent with one another. The operation of the compressor is dependent on the speed of the spool to which it is attached, and the speed of the spool is dependent on the torque 'absorbed' by the compressor. This interdependency is also present in the turbines, and is what dictates the rotor dynamics of the engine.

Finally, it is evident how easy it is to include more or fewer compressors within an engine model.

Combustion

The next component, after the compressors, on the primary gas path is the combustion chamber/s. The combustion process is surrounded by the high pressure turbine and compressor, as illustrated in Figure 5.10 [1]. However, during transient operation, there will occur a mismatch between the flow

[1]This figure does not directly correspond to Figure 5.4, as the turbine blocks were simply relocated closer to the combustion model, which is more visually representative of the gas path

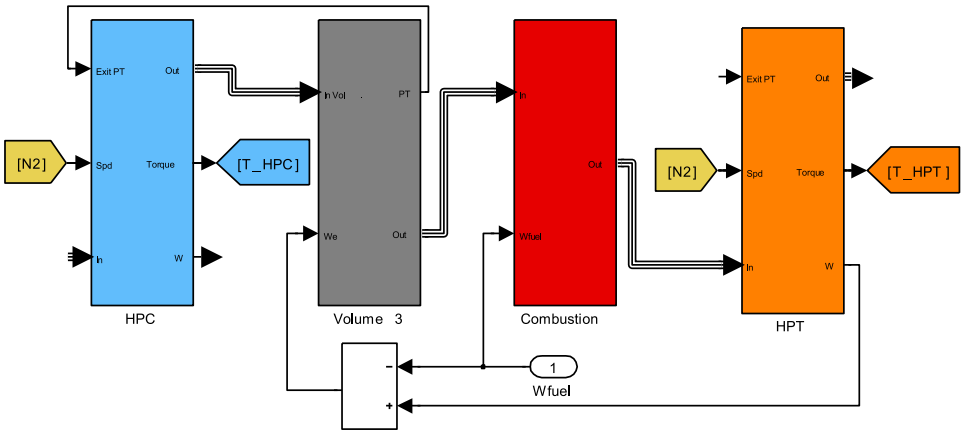


Figure 5.10: Combustion Simulink Block Assembly, zoomed in from Figure 5.4

rates of the compressor and turbine. Hence, a ICV has been placed in-between the high pressure compressor and the combustion model. This volume dictates the pressure at which combustion occurs, as well as the exit pressure of the high pressure compressors. Should any pressure loss occur during combustion, it is accounted for within the combustion model before the flow reaches the turbine. It is important to note that the fuel mass flow rate must be subtracted from the mass flow rate of the turbine when determining the amount of mass exiting the ICV. Apart from the above, the combustion process is only really dependent on the flow conditions from the compressor, as well as the fuel injected.

Turbines

The assembly of the turbines is so similar to that of the compressors, and therefore does not warrant an in-depth description. In addition, the turbine-nozzle coupling is also the same as the compressor-nozzle coupling described for the bypass gas flow, and also does not warrant further description. Nonetheless, the turbine blocks assembly, along with the primary nozzle, are illustrated in Figure 5.11. In addition, this figure also illustrates the calculation of the net thrust of the engine, which is a sum of the intake's ram drag and the thrust produced at the primary and secondary nozzles.

Spools

Finally, the spool components are depicted in Figure 5.12. This illustrates how the spool components take in the torques of its rotary components to evaluate the speed at which it operates. The communication between the spools and the rotary components, uses colour coded tags for signal routing.

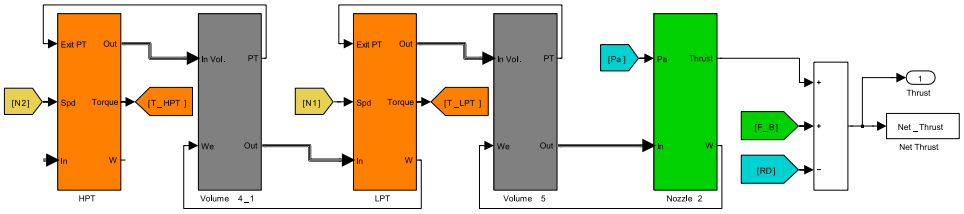


Figure 5.11: Turbine, primary nozzle and net thrust Simulink Block Assembly, zoomed in from Figure 5.4

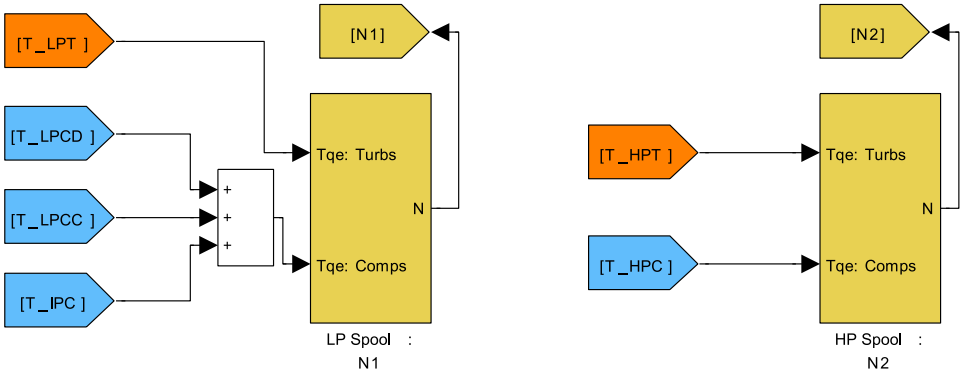


Figure 5.12: Spools Simulink Block Assembly, zoomed in from Figure 5.4

As is evident, the fan, intermediate compressor and low pressure turbine are all fixed to the low pressure spool. Whereas the high pressure compressor and turbine are fixed to the high pressure spool. These spool components are what dictate the rotor dynamics of the engine (i.e. acceleration and decelerations). It is evident here, that it becomes very simple to include additional or fewer spools (e.g. three spool turbofan or single spool turbojet engine).

5.3 Model Design Point Calculation

In the previous section, the assembly of a an engine model was discussed, more specifically, one that represents a twin-spool high-bypass turbofan engine. However in order for this model to become functional, its design operating point must be defined, which is then used as a reference for further simulations. This section discusses the evaluation of the design point parameters.

The user is not expected to supply all of the design point data for each component (the design point values have been discussed in Appendices D and E). In fact, if the user did supply all of this information, it would be very unlikely that the design point would be in a state of equilibrium. Instead, it is better that the user provide only some of the design point data,

leaving the model to calculate the remaining parameters. In this way, the engine model will be able to obtain a design point that is in equilibrium (i.e. no mismatch in torque or flow). The model configuration of Figure 5.4, has been specially adapted to allow for design point calculations, and is depicted in Figure 5.13. The most noticeable difference between Figures 5.4 and 5.13, is the absence of the ICV's. This is because the design point is in a state of equilibrium, and therefore there is no mismatch in flow between adjacent components, and no need to account for mass accumulation (i.e. the mass flow rate of each component is determined according to the mass flow rate of its preceding upstream component). In addition, the components of Figure 5.13 have been modified from those in Figure 5.4, to allow for the design point calculation. These modifications will be discussed along side the design point calculation process. Again, it is best to explain the design point calculation following the gas path from the intake to the nozzles. At this point it is also important to note that the design parameters required by the user, have been highlighted in red in Figure 5.13, as well as the remaining figures in this description..

Typically, the design point of an engine correlates to the steady-state operating point of highest efficiency at static sea-level conditions. The atmospheric pressure and temperature, at static sea-level conditions, is equivalent to standard atmospheric conditions (i.e. $T_{\text{STP}} = 288.15\ K$ and $P_{\text{STP}} = 101.325\ kPa$). Nonetheless, the user is first required to specify the altitude and airspeed of the engine's design point operation. These inputs are fed into the flight conditions block, which has remained unchanged, to calculate the local atmospheric conditions.

Next, the user must specify the design mass flow rate of the intake, which is the total flow of air ingested by the engine at this design point condition. Previously, in Figure 5.4, the intake's mass flow rate was taken to be equal to that demanded by the fan. In addition, the pressure recovery across the intake (for subsonic aircraft, this value is generally unity) must be defined inside the intake block (which has remained unchanged). These parameters define the total mass flow rate, stagnation pressure and temperatures entering the fan component.

The flow from the intake is then split into two streams. This split is defined by the design bypass ratio, which is a measure of the amount of bypass versus core flow:

$$BPR_{DP} = \frac{\dot{m}_{bypass}}{\dot{m}_{core}} \tag{5.4}$$

Therefore, the bypass and core flows may be obtained as a function of the design inlet flow and bypass ratio, given by Equations 5.5 and 5.6, respectively.

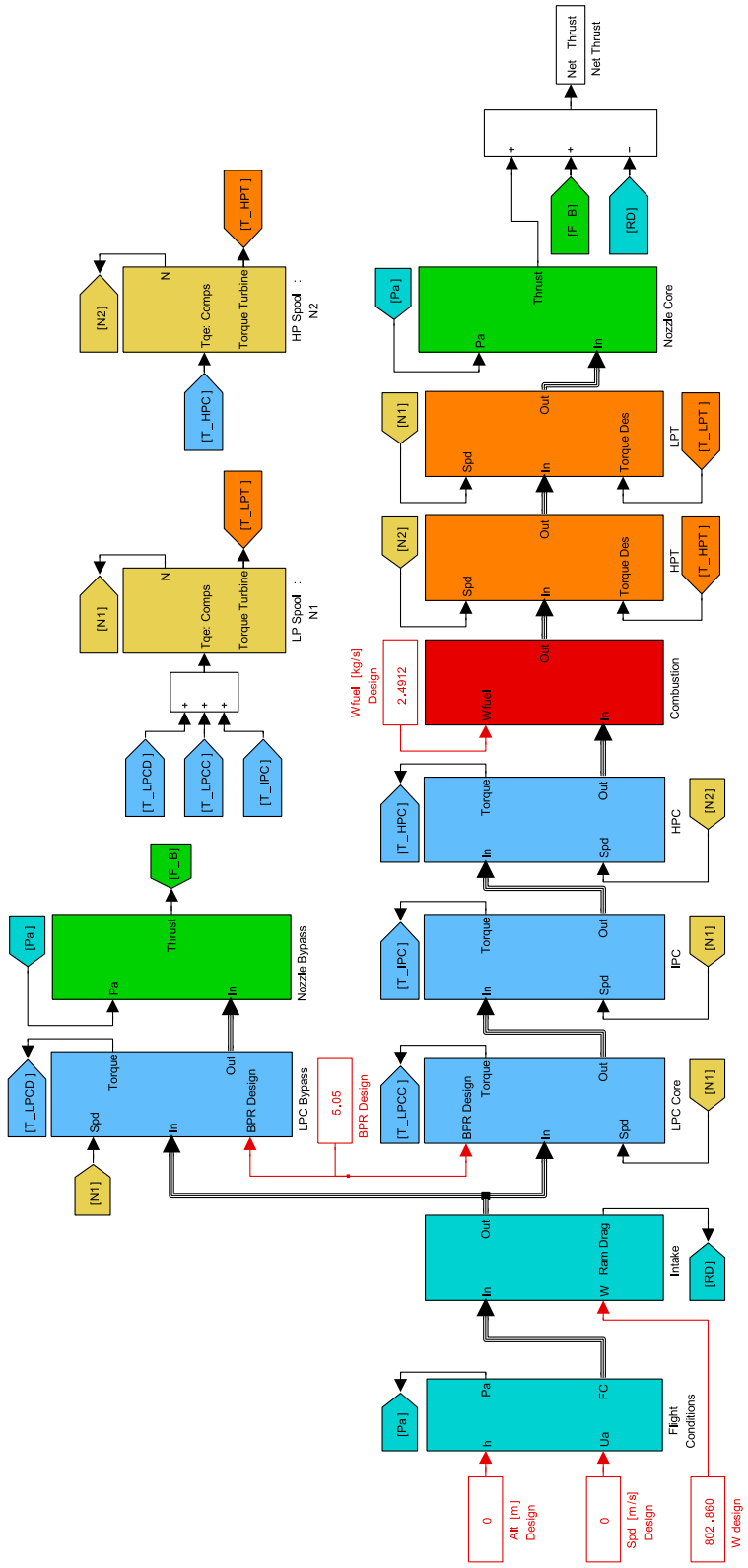


Figure 5.13: Twin-Spool High-Bypass Turbofan Engine Model fo Design Point Calculations in Simulink

$$\dot{m}_{bypass,DP} = \dot{m}_{intake,DP}\frac{BPR_{DP}}{BPR_{DP}+1} \tag{5.5}$$

$$\dot{m}_{bypass,DP} = \dot{m}_{intake,DP}\frac{1}{BPR_{DP}+1} \tag{5.6}$$

At this point the mass flow rate, pressure and temperature entering each of the fan's compressors (recalling that two parallel compressors have been used to simulate the radial profile of the fan) is known. It is now required to define the operating design point of each of these compressors.

The design point of a compressor defines the scaling of its characteristic map. There are five parameters which are required to scale a compressor map according to this design point; the map's β value[2], corrected operating speed N_C, corrected mass flow rate $\dot{m}_C$, isentropic efficiency η_c and pressure ratio PR[3].

The β value is normally kept the same as that of the unscaled map, or alternatively is close to 0.5 which often correlates closely with the compressors running line. The corrected operating speed and mass flow rate is calculated according to Equations D.1 and D.2 of Appendix D. To allow for the calculation of Equation D.1, the design spool speed must also be specified. Actually both spool's design speeds must be specified, which are in-turn used to define the design point of the rotary components. Finally, the values of each compressor's isentropic efficiency and pressure ratio are required. The two compressor components, making up the bypass and core of the fan, have been modified to include the above calculations, and this modification is depicted in Figure 5.14.

As already mentioned, these five parameters are used to scale the compressor's characteristic map according to that design point. Next, these design point parameters, knowing the compressor's inlet conditions, are used to determine the total pressure and temperature exiting the compressor. This calculation involves an iterative procedure, which has been detailed in Figure E.7 of Appendix E.3. Following the gas path through the core of the engine, this same process is repeated to determine the design operating point of the intermediate and high pressure compressor; who's inlet mass flow rate, temperature and pressure is equal to the exit conditions of its preceding component. The modifications to the remaining compressor blocks are very similar to those made to the fan's compressor blocks. The only difference is that the compressor block does not take in bypass ratio, and assumes a mass flow rate equal to that of its previous component. This difference is evident in Figure 5.15.

[2]The β value gives reference to an axillary coordinate system, which is used to aid the two-dimensional interpolation of compressor maps. This is detailed in Appendix D

[3]The corrected terms scale the map with reference to standard atmospheric conditions, this allows for the use of the same compressor map at different flight conditions (i.e. altitude and airspeed). This has been detailed in Appendix D

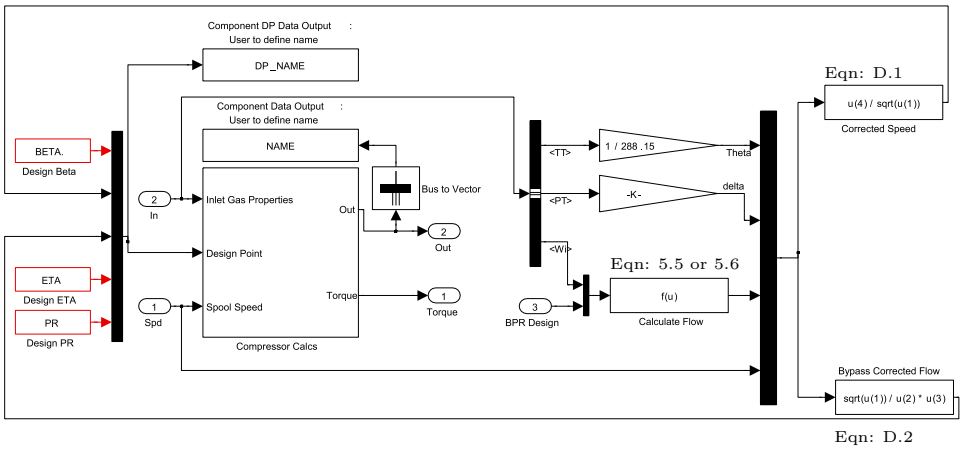


Figure 5.14: Modified design-point compressor block, for use in fan model

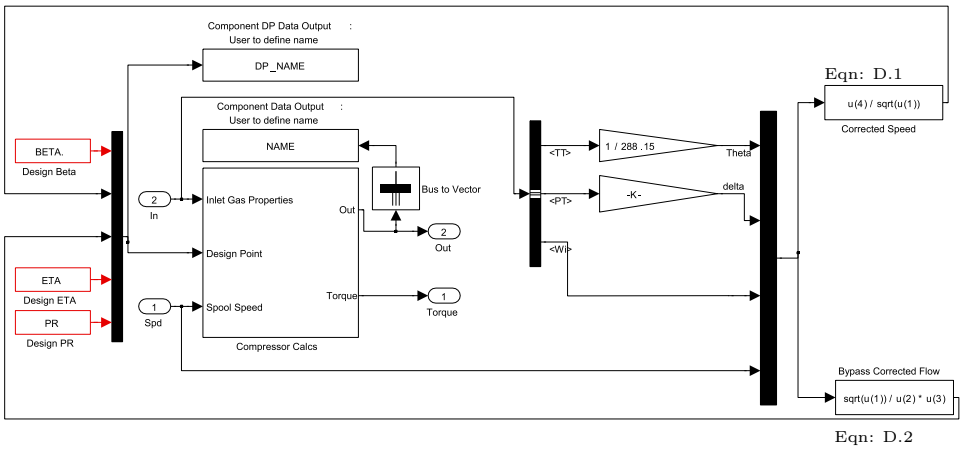


Figure 5.15: Modified design-point compressor block

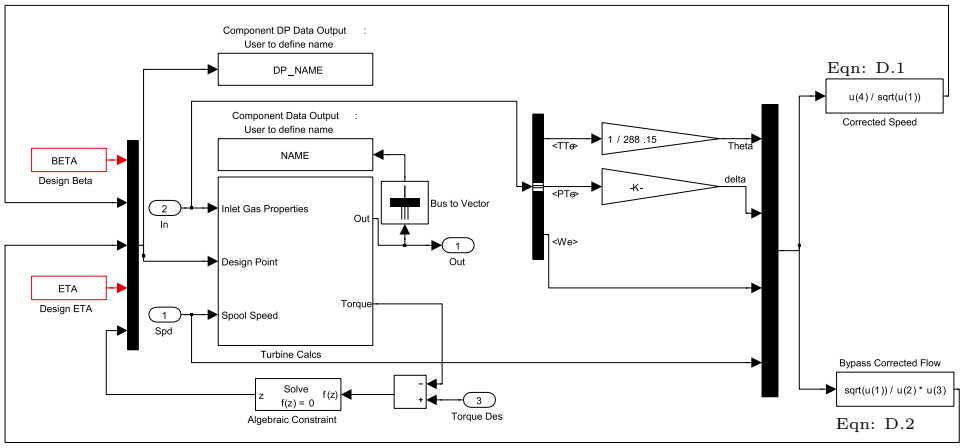


Figure 5.16: Modified design-point turbine block

The combustion process requires the fuel flow rate, fuel temperature and combustion pressure losses to be specified. These parameters, along with the input from the high pressure compressor, are used to calculate the properties of the combustion products, as has been detailed in Appendix E.5. No modifications were made to the combustion block. The flow conditions exiting the combustion process, subsequently, define the high pressure turbine's inlet conditions.

As with the compressor, the turbine also requires the same five parameters to scale the turbine's characteristic map, thereby defining its design point of operation. Again, the turbine's β value is to be user defined, and is generally kept the same as that of the original turbine map. The corrected speed and corrected mass flow rate, like with the compressor, is calculated from Equations D.1 and D.2 knowing its inlet conditions resulting from the combustion process. The isentropic efficiency is also user defined, however the pressure ratio is not. Instead, the pressure ratio must be calculated such that the torque produced by the turbine matches that of its corresponding compressor/s. This calculation must be performed iteratively, as the amount of torque generated is a function, amongst others, of the turbine's exit temperature (an iterative solution is required for this to account for variable specific heats). This matching of torque, also takes into account any losses due to mechanical inefficiencies (see Equation 5.2), another parameter which is user specified. Finally, once all five parameters have been defined, the exit conditions of the high pressure turbine may be determined, as detailed in Appendix E.6, and used in defining the design point of any subsequent turbines through a similar process. The modified turbine block is depicted in Figure 5.16, and although similar to the compressor block, it includes additional blocks to calculate the pressure ratio as already discussed.

The spool block has been modified so that it no longer calculates the

Component	Parameter	User-defined	Calculated
Intake	$\dot{m}$	Yes	
	PR	Yes	
Fan	BPR	Yes	
Compressors	β	Yes	
	N_C		Yes
	$\dot{m}_C$		Yes
	η	Yes	
	PR	Yes	
Combustion	$\dot{m}_f$	Yes	
	T_f	Yes	
	PR	Yes	
Turbines	β	Yes	
	N_C		Yes
	$\dot{m}_C$		Yes
	η	Yes	
	PR		Yes
Nozzles	η	Yes	
	PR	Yes	
	A		Yes
Spools	N	Yes	
	η	Yes	

Table 5.1: Engine model design point parameters

acceleration of the spools. Instead, its speed is fixed to the design point value specified. It is used to determine the turbine's torque based on the input torque from the compressor and a specified mechanical efficiency. Due to its simplicity, the modifications are not presented here in a figure.

It remains to define the effective area of the nozzles at this design point. Here, the same nozzle calculations are used as those presented in Appendix E.7. Except this time the flow rate through the nozzle is assumed to be equal to that of its preceding component. Therefore, after calculating the gas' density and exit velocity, the mass flow rate is used to determine the effective area of the nozzle, instead of the other way around. This is also a minor modification, and has therefore not been documented any further.

As a summary, Table 5.1 gives the parameters required for defining the design point of the engine. It indicates those parameters, which are to be user-defined, and those, which are to be determined from a design point calculation.

5.4 Transient Model Initialisation

The previous section discussed the purpose of the design-point calculations, required to make the model functional. However, additional information is still required in order to perform transient simulations. This is because the

transient model is comprised of a set of differential equations, describing the volume and rotary dynamics which define the engine's operation. These differential equations require initial solutions, or estimates thereof, to allow the model to function properly within the first time step [4]. Therefore, the turbofan engine model of Section 5.2, is described state vector X (Equation 5.7), initial values X_0 (Equation 5.8) and control inputs x (Equation 5.9).

$$\dot{X} = \left[\dot{N}_{LP}\ \dot{N}_{LP}\ \dot{P}_{2.1}\ \dot{P}_{2.2}\ \dot{P}_3\ \dot{P}_{4.1}\ \dot{P}_5\ \dot{P}_{18}\right] \tag{5.7}$$

$$X_0 = \left[N_{LP}\ N_{LP}\ P_{2.1}\ P_{2.2}\ P_3\ P_{4.1}\ P_5\ P_{18}\right]_0 \tag{5.8}$$

$$x = [\dot{m}_f \text{ Altitude Airspeed}] \tag{5.9}$$

This section discusses the calculation/estimation of the parameters in Equation 5.8, prior to a transient simulation. Ideally, this requires a steady-state engine model, which solves the engines states for a specified fuel flow rate, altitude and airspeed. At design conditions, the initialisation parameters (vector in Equation 5.8) are simply obtained from the design-point calculations model. However, the design-point is calculated at static sea-level conditions, but, as in the case of this work, the engine model is required to be initialised at some other flight condition. Simply altering the fuel, altitude and air speed inputs to the transient model (initialised using design-point parameters), will result in errors that will cause the model to crash. This is because the change in flight conditions (pressures and temperatures) will move the operating point outside the bounds of the compressor's and turbine's maps. For example, should the pressure ratio across a compressor far exceed the maximum for a particular running speed, the map interpolation procedure is required to perform an extrapolation. This extrapolation may result in extremely low mass flow rates, in some cases causing flow reversal. Thus an additional model is required to find a steady-state operating point (if it exists) within the compressor and turbine map bounds, correlating to a given fuel flow rate, altitude and airspeed.

Unfortunately, a steady-state version of the engine model presented in this work, has not yet been developed. However, the transient model may be used to perform steady-state calculations in a crude way. This may be done by slowly changing the flight conditions (fuel flow rate, airspeed and altitude) from the design point condition to the required flight condition. After allowing for sufficient time for the transient model to settle, thereby

[4]Incorrect initialisation may start the engine model at an operating point that won't converge on a solution. For example, if the pressure ratio of the HPC is too low, a very small mass flow rate of air will be passed to the combustion chamber. This may result in too low an air-fuel ratio for successful combustion. Alternatively, in another example, an ill initialised model may result in dynamics that are too violent, thereby far exceeding compressor map bounds and causing flow reversal

converging on a steady-state solution, the final results of the simulation may be saved and used to initialise future instances of the transient model at those conditions.

5.5 Engine Control

As with many dynamic systems, the engine requires a controller to function effectively. Control systems for commercial turbofan engines are very complex systems. The purpose of these control systems is to ensure that the engine operates safely and efficiently. To that end, the control system must prevent compressor surge, spools over-speeding, combustion flame-out, nozzle choke and turbine-blade over-heating.

Various systems have been developed to deal with such issues. Variable-inlet guide vanes are used to prevent compressor surge by altering the effective geometry of the compressor blades to suit different flow conditions. Compressor bleeds are used to cool turbine blades, thereby increasing the allowable turbine inlet temperature which in-turn improves thrust and engine efficiency. However, one of the most important control systems is that of fuel metering, as it is the throttle that ultimately dictates the operation of the engine.

As one can imagine, the control system of a modern turbofan engine is very complex, and therefore its full inclusion within the model developed in this work has been omitted. Instead, a very simple and conservative approach has been taken to control the engine's functionality via fuel metering. Therefore other systems, such as compressor bleeds and variable inlet guide vanes, have been ignored.

When controlling the fuel flow rate to the engine, it is important to consider a number of factors. Firstly, assuming steady-state operation, there will exist a maximum and minimum fuel flow rate. Exceeding these limits may result in combustion flame-out, as a result of either too low a concentration in oxygen or fuel, respectively. In addition, the fuel flow rate dictates the speed at which the engine operates, and the rotary components have a limits on the speed at which they can operate.

Secondly, it is important to consider the transient effects of changing the fuel flow rate. Once again, combustion flame-out may result from too large a change in fuel in too short a time. This is because the permissible fuel-air ratio limits, which allow for successful combustion, are exceeded. In addition, large and/or fast throttle changes may cause the rotary components to exceed their operating limits. As an example, a drastic increase in fuel results in a sudden increase in pressure ratio across the compressors,and occurs well before the spools have had time to react. Therefore, the compressors operating points will shoot up its current speed line, thereby exceeding the pressure ratio and mass-flow rate at which surge occurs. In

actuality, the fuel metering control system needs to take into account many interdependent factors, and is therefore also difficult to model.

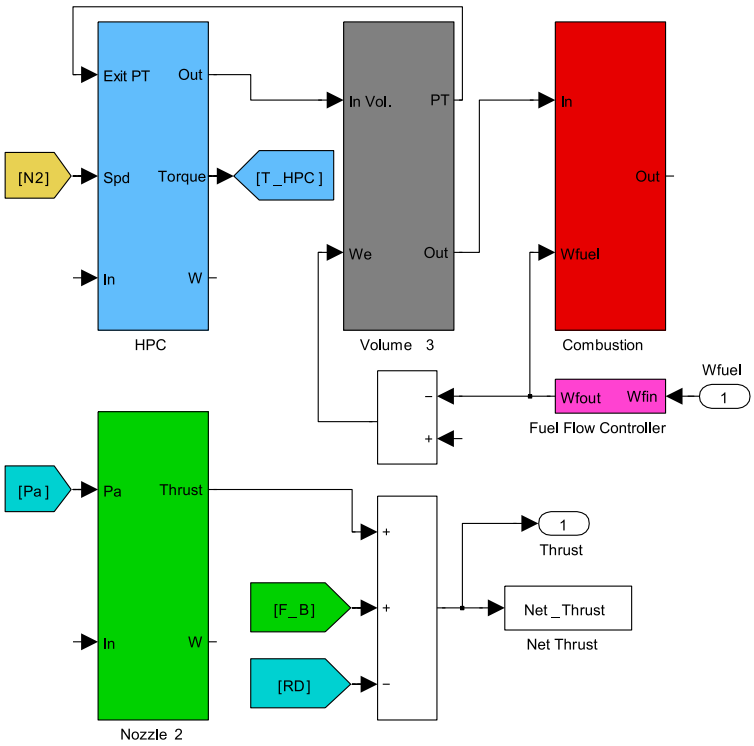
However, a simple approach to fuel metering may be taken, to ensure that the engine model does not enter a point of operation which causes it to crash (i.e. the same operating points at which the engine would fail to function normally). The first is to set a limit on the the maximum and minimum fuel flow rates that are metered to the combustion chamber. Secondly, limits can be placed on the rate at which the fuel flow may be altered. Such slew rate limits will differ according to the current operating point of the engine, the required fuel change and whether the engine is accelerating or decelerating. Furthermore, there may exist a change in permissible slew rate as the engine accelerates. This in-turn becomes quite difficult to model, however further simplifications may be made to circumvent such complexities.

For a given flight condition (i.e. altitude and airspeed) it is easy to use the model to determine the maximum and minimum fuel flow rates, by slowly altering it until the engine model crashes. These limits may then be used to determine an overall maximum rising and falling slew rate, which is assumed to be linear and independent of the engine's operating point. It is important to note that both an accelerating and decelerating slew rate limits must be defined independently, due to the hysteresis in the engines dynamic path. As in the case of the total fuel flow rate limits, the slew rate limits are found via trial and error, by establishing the fastest rate at which the fuel may be throttled from its minimum to maximum fuel flow rates (and vica versa) without the engine model crashing. This is a rather crude approach, however it will ensure that the engine model will always function despite any 'reckless' throttling by the aircraft model. In addition, the development of a truly representative engine control system is too complex for the scope of this work.

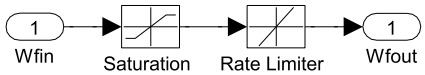
The simple fuel controller used in this work, is depicted in Figure 5.17. Figure 5.17a illustrates its position within the engine model assembly of Figure 5.4. Figure 5.17b illustrates the contents of this block, which include a Simulink saturation and rate limiter for defining the maximum/minimum fuel flow and slew rates respectively.

5.6 Numerical Solver

Gas turbines dynamics, as presented thus far, are primarily modelled as rates of change in pressures and spool speeds. These dynamics are in the form on a set of non-linear ordinary differential equations, which has been denoted in Equation 5.7. The volume dynamics (rates of change in pressure) occur far more rapidly than the rotor dynamics. Therefore, the overall system is stiff, and requires an appropriate variable-step solver. Using a fixed step solver requires an extremely small time step and results in an impracticably



(a) Fuel flow controller block (highlighted in magenta) positioning within engine model assembly of Figure 5.4



(b) Fuel controller subsystem with sturation and slew rate blocks

Figure 5.17: Engine model: Fuel controller developed in Simulink

inefficient and slow simulation. The MATLAB/Simulink environment has many built-in solvers, which include some which are adept at solving stiff problems.

All of the MATALB/Simulink solvers, appropriate for stiff systems, were successfully applied to the engine model developed in this work. However, the 'ode15s' provided the highest efficiency, for the same level of accuracy as the others [5], and is based on Gear's Method [112]. It is noted, that this solver was applied to the engine model in an isolated scenario, and that a different solver may be more suited to the case where the engine is included within an aircraft model within a formation model.

5.7 Validation

The generalised twin-spool, high-bypass turbofan configuration, described in Section 5.2, was used to represent General Electric's CF6-80C engine. This particular engine was chosen as GSP provides a sample model ('BIGFAN') of a turbofan engine, which is claimed to be representative of the CF6-80C [113]. In addition, this engine may be found on some versions of Boeing's 747, which is the aircraft upon which this work is based. Therefore, engine data and information of GSP's BIGFAN sample model (representative of the CF6-80C), was applied to the engine model developed in this work. The model's simulations were then compared to the BIGFAN model of GSP, in order to validate this work. Thereafter, if valid, the model developed in this work is too deemed to be representative of the CF6-80C.

The purpose of the original GSP BIGFAN model, was to predict steady-state performance, and therefore did not include any ICV's. As is recommended in the GSP user manual [113], the GSP 'Duct' component should be incorporated between components to model volume dynamics (i.e. the 'Duct' component is similar to the ICV component of this work). Therefore, 'Duct' components were included to replicate the configuration depicted in Figure 5.3.

As with the ICV's developed in this work, the 'Duct' models required information as to the size of the volumes located between components. In addition, value as to the spool inertias were not provided. Unfortunately, because manufacturers keep most information proprietary, these values were not available to the author. Estimations of the volume sizes (H. I. H. Saravanamuttoo, Professor at Carleton University. Personal communication, 14 March 2013) and spool inertias (D. Parker, Aero-thermal Engineering Manager at Chromalloy ECOE. Personal communication, 25 July 2013) were obtained from personnel in the propulsion industry, and have therefore been respectfully omitted from this report. It is noted that the values used to

[5] In fact, the Simulink documentation claims this solver to be the most accurate of the solvers appropriate for solving stiff problems

model the volumes and spools, are estimations and not exact true values, but are still sufficiently representative of a typical commercial high-bypass twin-spool turbofan engine. Therefore these estimations were applied to both models for the validation tests. The remainder of this section discusses the results of the validation tests.

5.7.1 Design Point

The design point calculations were described in Section 5.3. This section aims to validate those calculations against GSP's model described above. Table 5.2, gives the user-defined design-point parameters taken from the GSP model. These same values were input into the design-point model developed in this work, and a comparison of the results are presented in Table 5.3.

Table 5.3, indicates a close correlation between the two models. The error in the design-point calculated parameters does not exceed 5 %, and is generally found to be much lower than that. It is noted that the error tends to increase for those components located further downstream. This is attributed to the fact that each component is reliant on the results of its predecessor upstream, and therefore the errors start to compound on top of one another. In addition, the combustion model is not as sophisticated as GSP's, and therefore introduces some errors into the results [6]. However, the errors have a minimal effect on the overall thrust produced by the engine, which is well below 1%. Therefore, the design-point model developed in this work is deemed valid in comparison to GSP's.

5.7.2 Steady-State

Although the current model does not include a steady-state calculator, the transient model may be used to perform steady-state predictions. In this case, the input parameters (i.e. fuel flow rate, airspeed and altitude) are slowly brought from design point values to the those of the desired operating point. Those parameters are then held constant, whilst allowing the model's the dynamics to settle on a point of equilibrium. Once equilibrium is reached, the models parameter values represent the engine's steady-state operation for the specified flight conditions.

Unfortunately, depending on the desired flight conditions, this process may take quite some time. Nonetheless, this technique was used to compare steady-state predictions against results from GSP's model. The graph in Figure 5.18 illustrates the correlation in predicted thrust between GSP's model and the model developed in this work. This comparison was conducted at both static sea-level and cruise conditions for a range of different

[6] The errors in the combustion model have been discussed extensively in Appendix E.5

Component	**Parameter**	**Units**	**User-defined Value**
Flight Conditions	Altitude	m	0
	Air Speed	m/s	0
Intake	$\dot{m}$	kg/s	802.860
	PR	-	1
Fan	BPR	-	5.05
LPC Bypass	β	-	0.571429
	η	-	0.93
	PR	-	1.7
LPC Core	β	-	0.571429
	η	-	0.93
	PR	-	1.65
IPC	β	-	0.571429
	η	-	0.91
	PR	-	1.495
HPC	β	-	0.60979
	η	-	0.840
	PR	-	11.980
Combustion	$\dot{m}_f$	kg/s	2.4912
	T_f	K	288.15
	PR	-	0.96
HPT	β	-	0.864906
	η	-	0.920
LPT	β	-	0.700
	η	-	0.900
Bypass Nozzle	η	-	0.9506
	PR	-	0.9720
Core Nozzle	η	-	0.9506
	PR	-	0.9850
HP Spool	N	rpm	10300
	η	-	0.99
LP Spool	N	rpm	3390
	η	-	0.99

Table 5.2: Engine model user-defined design-point parameter values

Component	Parameter	Units	GSP	DP Model	Error (%)
LPC Bypass	N_C	rpm	3390	3390	0
	$\dot{m}_C$	kg/s	670.1559	670.1559	-4.8095×10^{-6}
LPC Core	N_C	rpm	3390	3390	0
	$\dot{m}_C$	kg/s	132.70414	132.70413	-5.8541×10^{-6}
IPC	N_C	rpm	3140.4	3140.4	-2.1951×10^{-4}
	$\dot{m}_C$	kg/s	86.86566	86.81851	-0.0543
HPC	N_C	rpm	8963.4	8963.4	-3.7074×10^{-4}
	$\dot{m}_C$	kg/s	61.90331	61.81947	-0.1354
HPT	N_C	rpm	4573.1	4576.3	0.0707
	$\dot{m}_C$	kg/s	11.14705	10.72578	-3.7792
	PR^{-1}	-	4.0641	4.0148	-1.2121
LPT	N_C	rpm	1747.4	1745.1	-0.1297
	$\dot{m}_C$	kg/s	38.74253	37.16613	-4.0689
	PR^{-1}	-	4.3874	4.3443	-0.9820
Core Nozzle	A	m	0.63204	0.61859	-2.1270
Bypass Nozzle	A	m	1.90928	1.90990	0.0326
Thrust		kN	254.77849	256.43063	0.6485

Table 5.3: Design-point model validation: compared against GSP

fuel flow rates. As is evident, there is a fairly close correlation between the two models (generally within 2%) except at extremely low fuel-air ratios.

The off-design steady-state operating lines (corresponding to Figure 5.18) of each component are depicted in Figures 5.19 to 5.24. Each of these figures illustrates the steady-state running line at a) static sea-level conditions and b) cruise conditions.

The running lines for the low pressure bypass compressor, Figure 5.19, illustrates an excellent correlation between the two models at both flight conditions.

The core low pressure compressor's running lines, Figure 5.20, are also correlate closely at both flight conditions. However, there is a tiny discrepancy between the running lines of the two models, which becomes slightly more evident at cruise conditions (Figure 5.20b) than at static sea-level conditions (Figure 5.20a). Furthermore, the running lines of both models do not pass directly through the design point, particularly at cruise conditions. This is expected at cruise conditions, as the design point is defined and corrected for static sea-level conditions, and therefore some slight errors are introduced when shifting to another flight condition.

Figure 5.21 illustrates the calculated running line of the intermediate compressor, which appears to suffer the least accuracy of all the compressors. This error, as with the low pressure core compressor, is worse at cruise conditions that at static-sea level conditions. It is suggested that the inter-

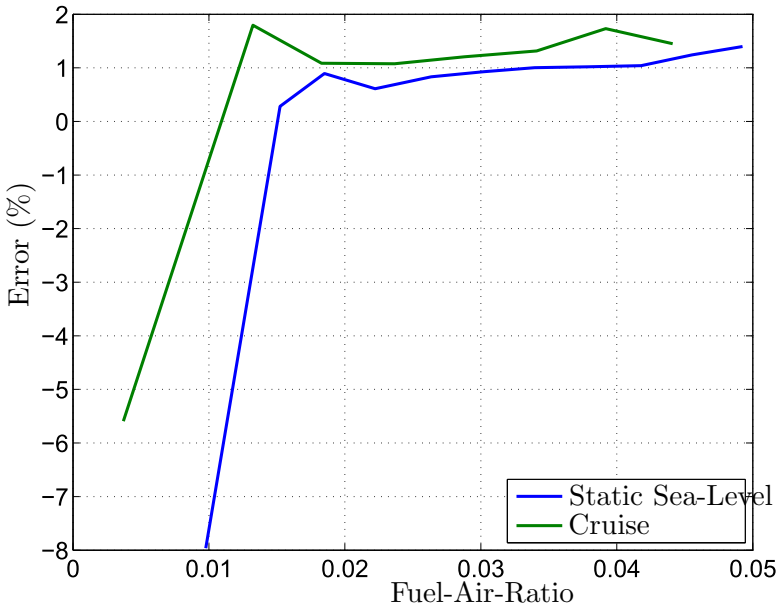


Figure 5.18: Validation of transient model's calculation of steady-state thrust

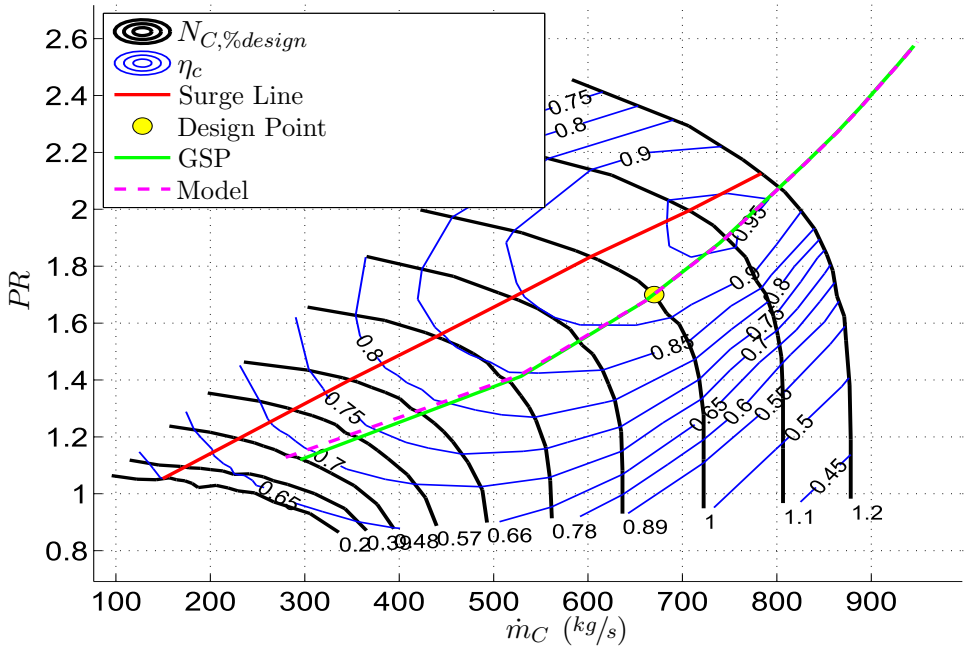
mediate compressor has higher errors than the low pressure compressors, as it inherits their errors which are compounded on top of its own.

It is also suggested, that the intermediate compressor exhibits less error than the high pressure compressor, because the high pressure compressor is coupled to a separate spool, with fewer components. Fewer components mean that each component does not have to compromise its optimum operating point (for a given speed) as much, when converging on a single collaborative solution. However, as will be explained later in the transient validation section, the spool speed suffers accuracy, as there are fewer components that influence its speed.

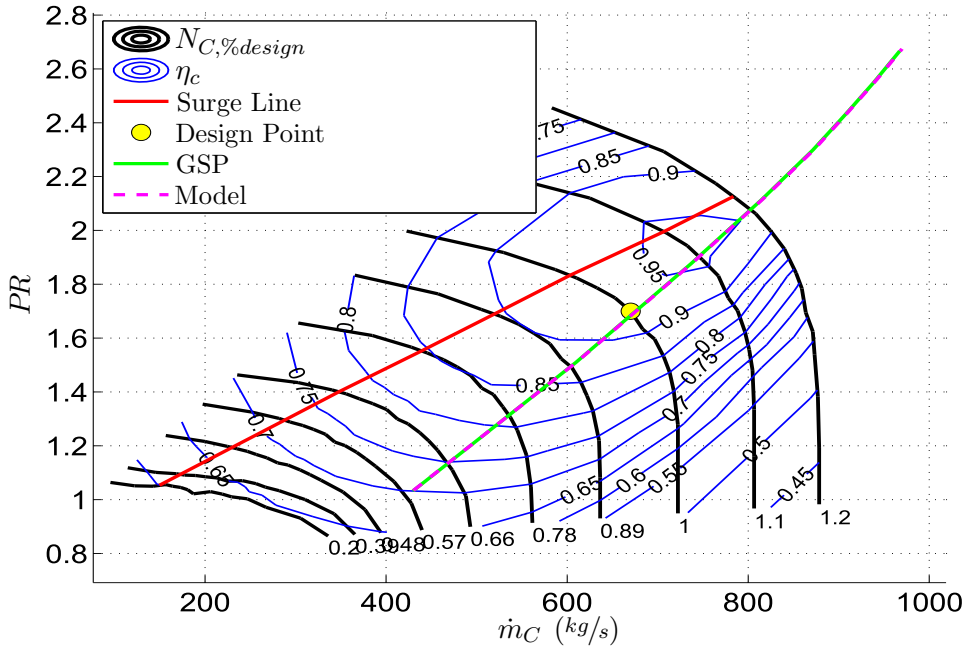
As already mentioned, the high pressure compressor running lines exhibits a close correlation between the two models, Figure 5.22b. It is of particular importance that the high pressure compressor model yields accurate results, as this determines the flow properties entering the combustion. Therefore, the more accurate the results from the high pressure compressor, the more accurate the combustion process and overall prediction of the engine's performance.

Figure 5.23, illustrates the running line of the high pressure turbine. It is clear from this figure that the turbine operates at a fairly constant pressure ratio. In addition, it operates within a very limited bandwidth of corrected speed and mass flow rate. Therefore, the correlation between the two model's running lines are close.

The low pressure turbine, on the other hand, is subjected to a much

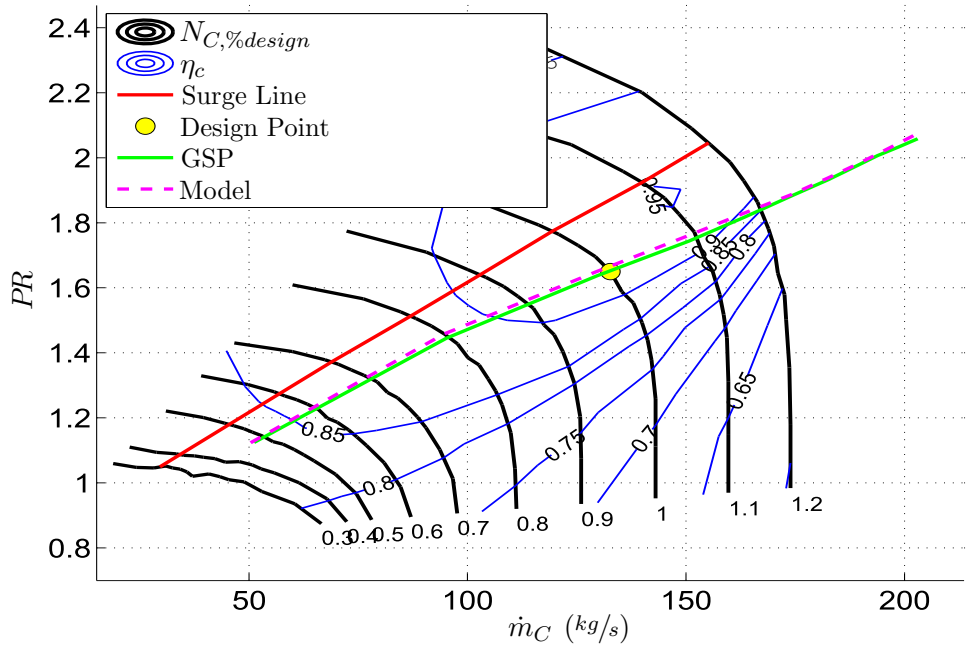


(a) Static sea-level conditions

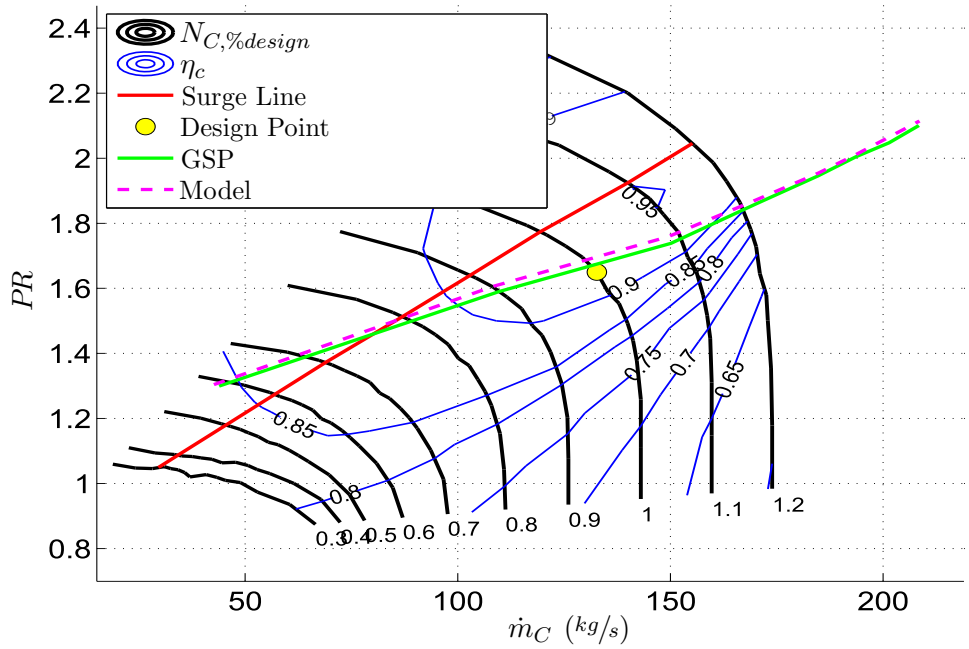


(b) Cruise conditions

Figure 5.19: Engine model validation: Comparison of low pressure compressor (bypass) off-design steady-state running lines

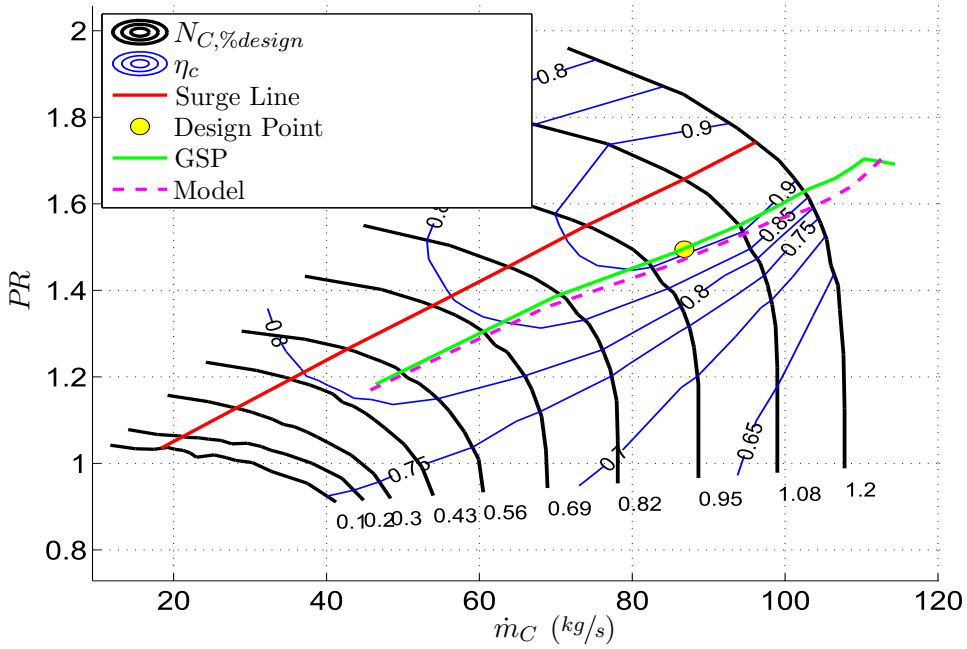


(a) Static sea-level conditions

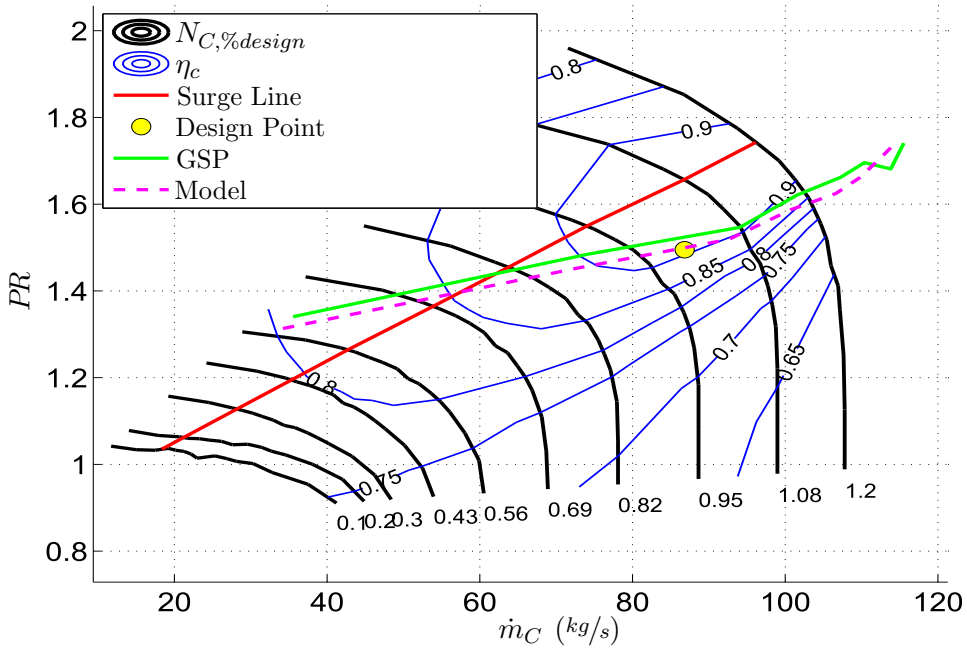


(b) Cruise conditions

Figure 5.20: Engine model validation: Comparison of low pressure compressor (core) off-design steady-state running lines

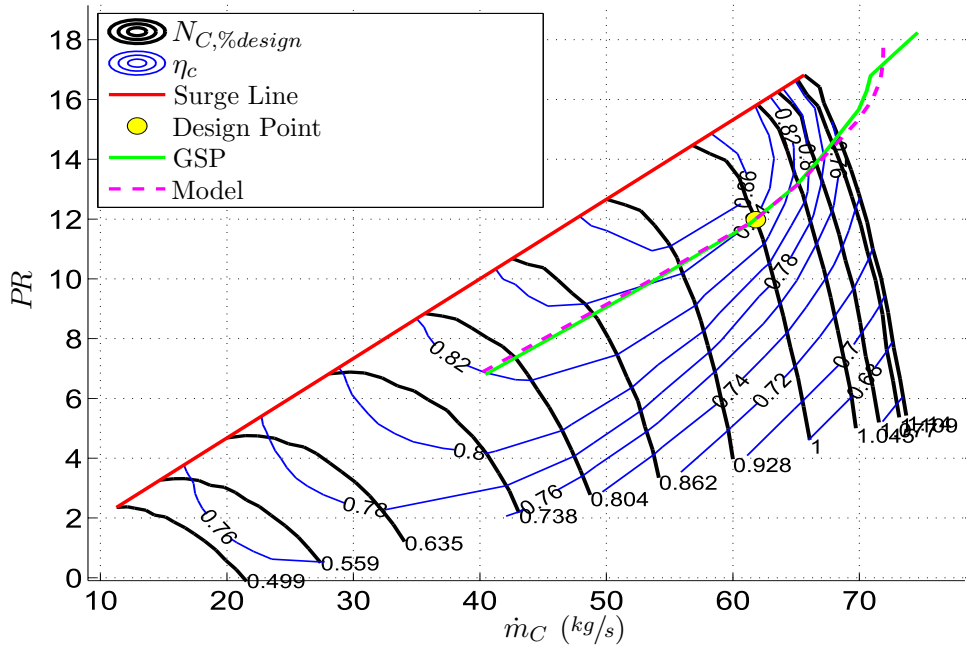


(a) Static sea-level conditions

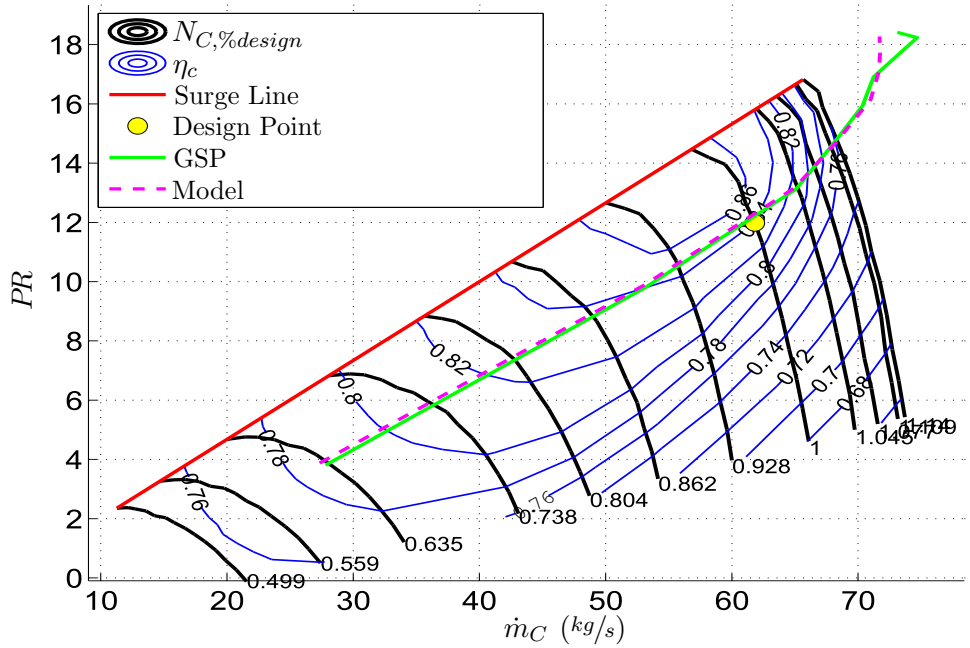


(b) Cruise conditions

Figure 5.21: Engine model validation: Comparison of intermediate pressure compressor off-design steady-state running lines

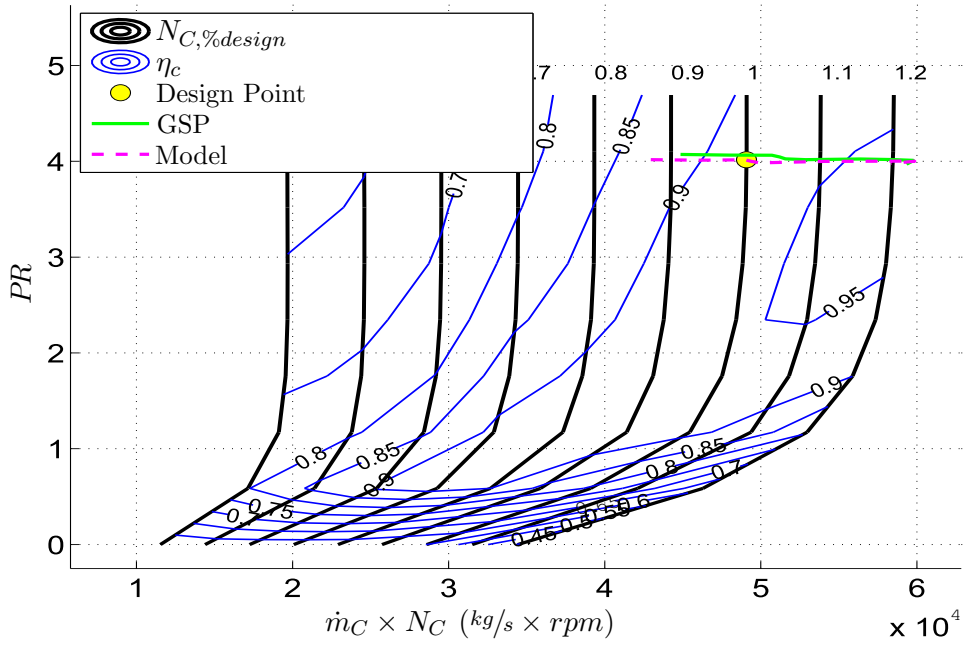


(a) Static sea-level conditions

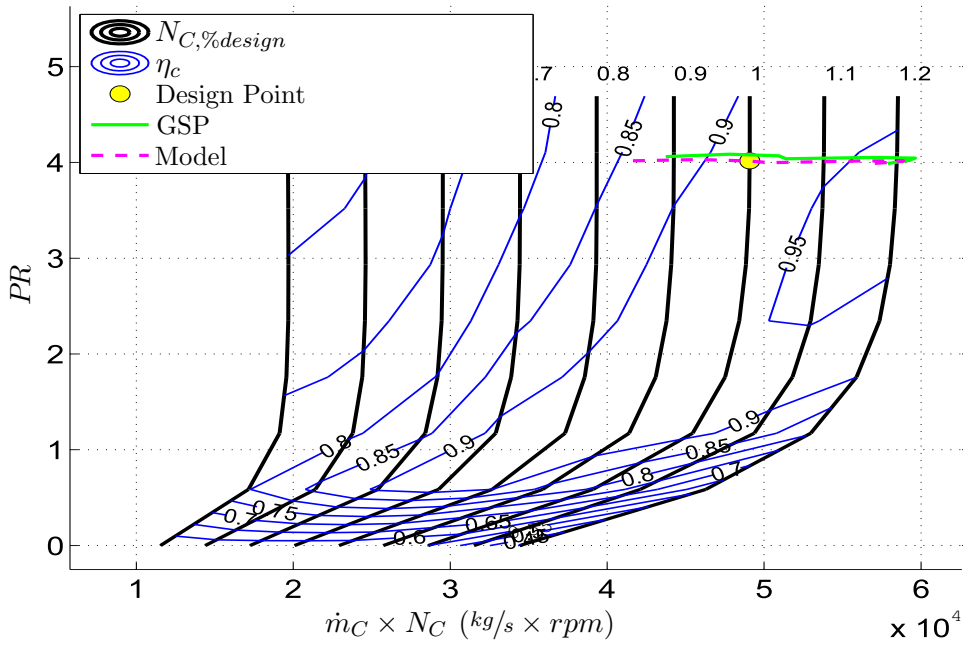


(b) Cruise conditions

Figure 5.22: Engine model validation: Comparison of high pressure compressor off-design steady-state running lines

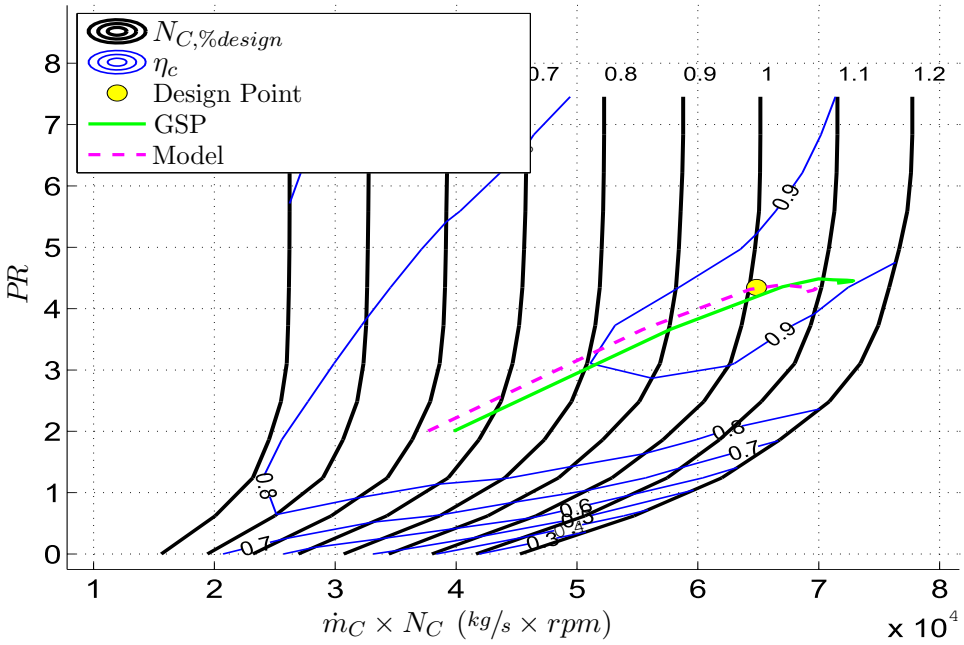


(a) Static sea-level conditions

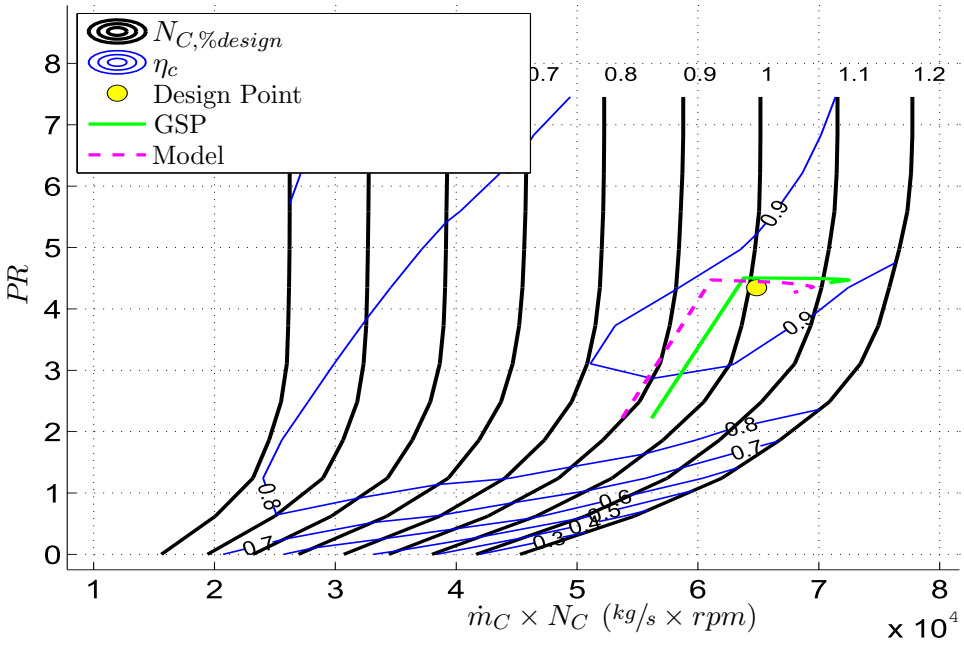


(b) Cruise conditions

Figure 5.23: Engine model validation: Comparison of high pressure turbine off-design steady-state running lines



(a) Static sea-level conditions



(b) Cruise conditions

Figure 5.24: Engine model validation: Comparison of low pressure turbine off-design steady-state running lines

larger operating region. Therefore, it is more becomes more vulnerable to errors, which is particularly evident at cruise conditions rather than at static sea-level conditions.

In general however, the correlation between the two models is sufficiently close to deem the model valid for predicting steady-state calculations at various flight conditions.

5.7.3 Transient

The purpose of the engine model developed in this work, is to accurately predict the transient performance of the engine for varying fuel commands. Therefore, it is very important to validate the engine model under dynamic conditions. The validation tests were performed at both static sea-level and cruise operating conditions.

Static Sea-Level

It is conventional to test the step response of a dynamic system during validation. However, both this work's and GSP's model are highly sensitive to a step change in fuel, when ignoring the fuel flow controller (see Section 5.5), and this sensitivity causes both models to crash. Therefore a different approach was adopted.

During the steady-state validation, in the previous section, the maximum and minimum allowable fuel flow rates were found and tested. Therefore, a slam acceleration from the minimum to maximum fuel flow rate was used as the transient validation test. The slam acceleration made use of a ramp function, to linearly increase the fuel flow rate from its minimum to maximum value. The shortest permissible time for this slam acceleration was found by trial and error, such that the engine model did not crash. Thereafter, it was used to define the rate at which the ramp function changed.

The engine was initiated at its design point. The initial fuel flow rate however, was set at its minimum value of 0.5 kg/s. Thereafter, the engine model was allowed to decelerate from the design point to a new steady-state operating point, which occurred within 15 seconds. Next, the slam acceleration was initiated and the fuel flow rate was increased linearly from its minimum value of 0.5 kg/s to its maximum value of 10 kg/s in 2 seconds. The engine model was then allowed to settle on a new steady-state operating point correlating to this maximum fuel flow rate value. The overall simulation took 30 seconds.

Figures 5.25 to 5.28, illustrate a comparison of this work and the GSP model's transient trajectories on each of the compressor's characteristic map. As is evident from theses figures, there is a good correlation between the two models.

The low pressure bypass compressor, Figure 5.25, shows a particularly

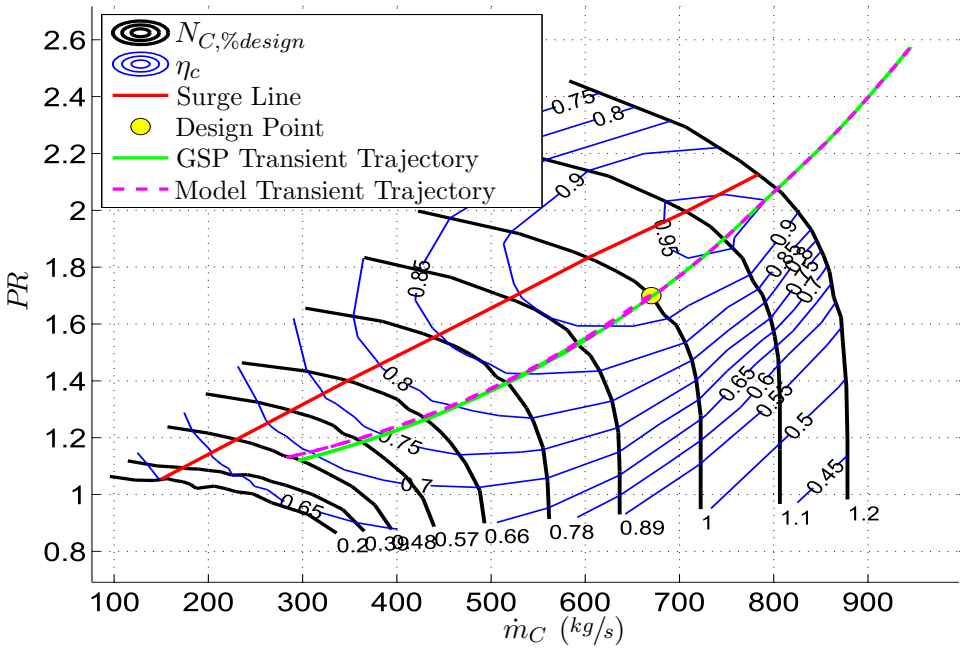


Figure 5.25: Engine model transient validation at static sea-level conditions: Low pressure (bypass) compressor map trajectories

close correlation between the trajectories of the two models. It appears as though the transients, for this particular compressor, move along its steady-state operating line (running line). This makes sense, because the bypass compressor's flow dynamics are independent of almost all of the other engine components. In fact, this compressor has very few constraints dictating its flow, because its boundary conditions are only restricted by the intake and nozzle. Furthermore, the intake's operating condition is somewhat dictated by this compressor's operating point, and thus the only flow constraint on this compressor is the operation of the bypass nozzle (which is not that restrictive). Therefore, the only real constraint placed on this compressor's operation, is the spool speed at which it operates. Apart from the operating spool speed constraint, it is relatively free to operate at the mass flow rate and pressure ratio of 'least resistance', which is the point of highest efficiency for that particular speed. Effectively, its flow dynamics adjust almost instantaneously according to changes in spool speed, and this therefore correlates to a operating point on the compressor's running line.

Figure 5.26, on the other hand, illustrates the transient trajectories of the core portion of the fan. Here, and as will be seen with the remaining compressors, flow dynamics are very dependent the operation of not only its adjacent components, but all of the components in its gas path. This is because there is a 'pushing and pulling' of the flow between the components (particularly the rotary components) as they all strive to reach their favoured

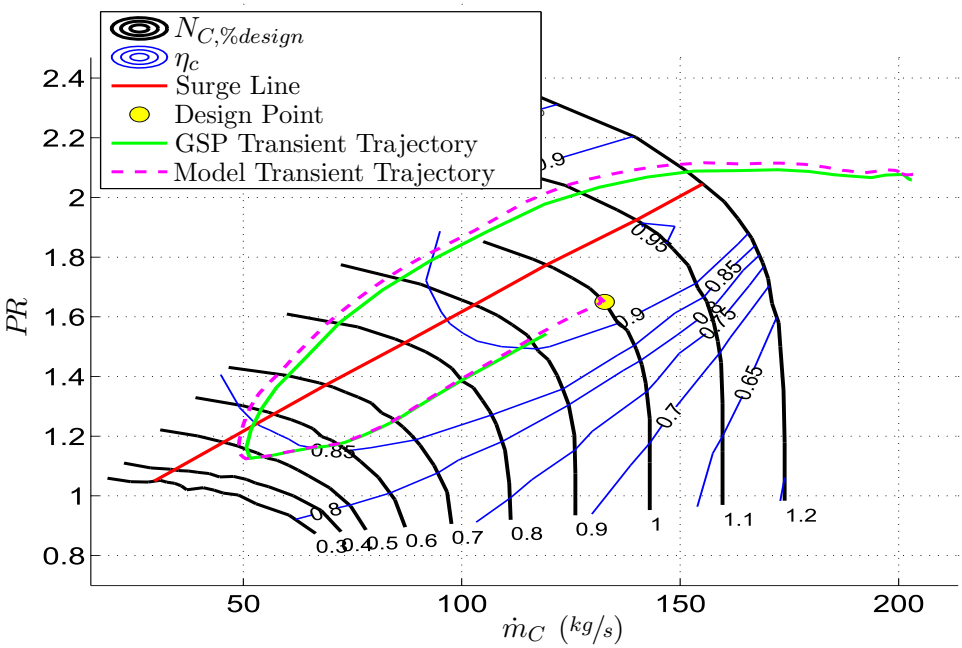


Figure 5.26: Engine model transient validation at static sea-level conditions: Low pressure (core) compressor map trajectories

operating point at each given speed. Thus the transient trajectories tend to deviate far away from the compressor's running line. In this map, one can see clearly how the engine decelerates from its design point to an operating point which correlates to the minimum fuel flow rate. Once the slam acceleration begins, the trajectory changes direction. Here, it is clearly visible how the change in spool speed (rotor dynamics) battles to keep up with the change in pressure ratio (volume dynamics), resulting in a trajectory that moves 'up' each speed isoline, away from the compressor's running line. Thereafter, as the spool gathers enough speed, the trajectory is able to move 'down' each speed isoline, back towards its running line. For this particular compressor, there is a slight discrepancy between the two trajectories during the slam acceleration. This is assumed to be a result of errors inherited from the design point model. However, as will be shown later, this discrepancy does not significantly effect the overall prediction on the engine's performance.

Figure 5.27, illustrates the trajectories superimposed onto the intermediate compressor map. The trends in the trajectories are the same as those mentioned for Figure 5.26, however now the discrepancy has increased somewhat. This increase may perhaps be a simple compounded combination of its own errors and that of its predecessor, as well as any errors that have occurred in the low pressure turbine. In addition, as mentioned in the analysis of the steady-state results, the intermediate pressure compressor is the third compressor coupled to the low pressure spool. The low pressure spool carries

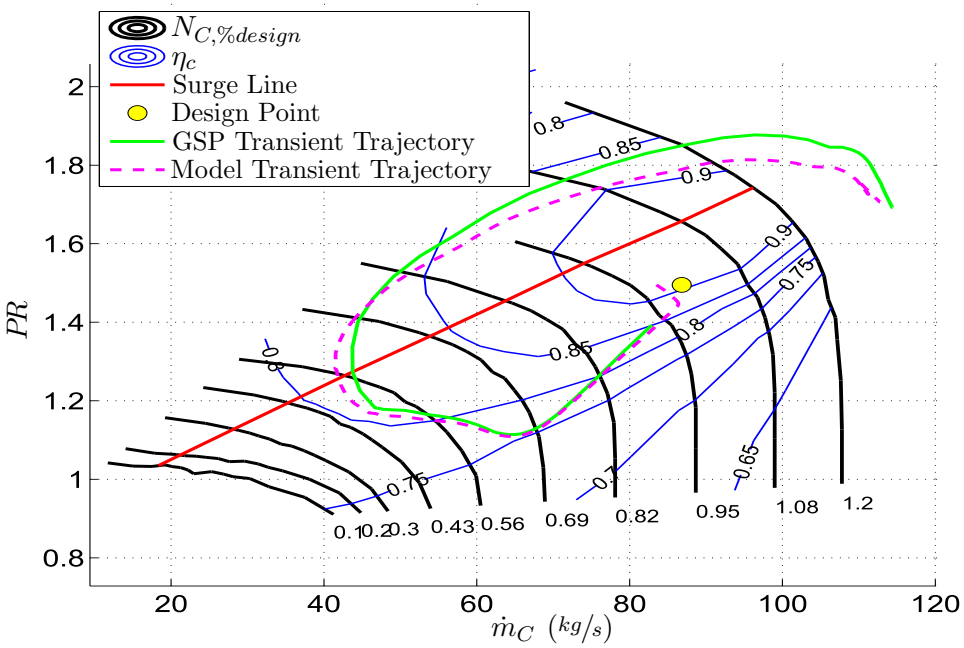


Figure 5.27: Engine model transient validation at static sea-level conditions: Intermediate pressure compressor map trajectories

many components, and therefore errors are introduced when these components all have to compromise their optimum operating point to converge on a single solution.

The two trajectories of the high pressure compressor (Figure 5.28), very importantly, correlate extremely closely. These trajectories illustrate perfectly the difference in speed between the rotor and volume dynamics. As the engine begins to decelerate from its design point, the trajectory can be seen to drop down directly along its speed isoline. This is because the flow adjusts itself to reach equilibrium long before the high pressure spool has had a chance to react. Thereafter the spool decelerates, whilst incrementally moving back 'up' each speed isoline towards its running line. Upon reaching this point, the slam acceleration is initiated, and the trajectory moves directly 'up' along its speed isoline, toward the surge line, again long before the high pressure spool has had a chance to react. Thereafter, as the rotor dynamics 'catch up', and the trajectories gradually make their way back towards the running line. The high pressure compressor correlations are more accurate because fewer components are coupled to the high pressure spool.

Figures 5.29 and 5.30, illustrate the transient trajectories of the high and low pressure turbines, respectively. It is not as intuitive to interpret these maps, because of the x-axis' definition. In these maps, unlike with the compressors, the trajectories don't start at the design point. This is because although the engine model's parameters were initialised at the design point,

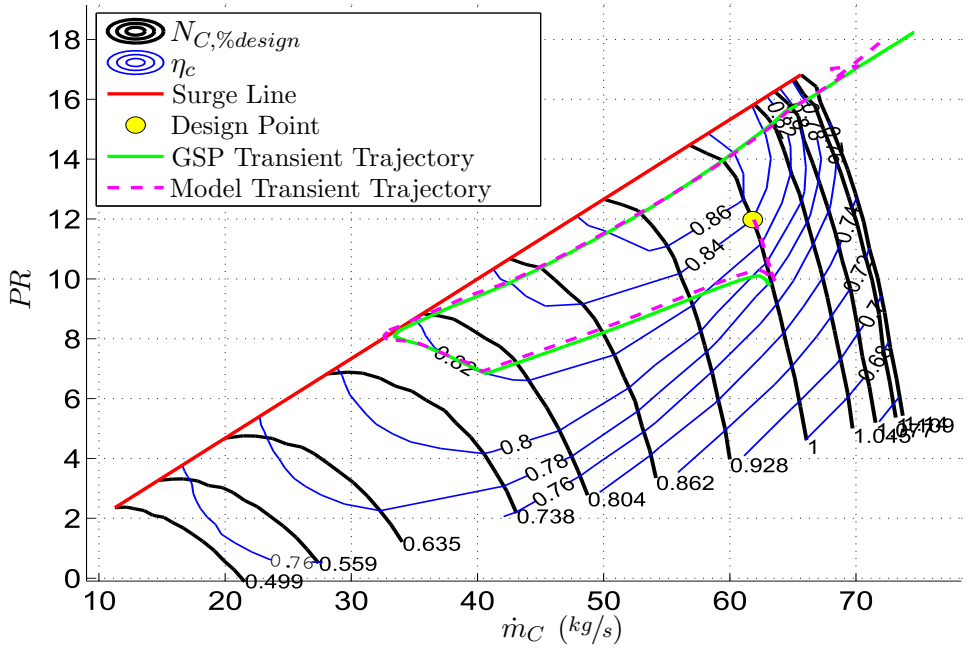


Figure 5.28: Engine model transient validation at static sea-level conditions: High pressure compressor map trajectories

the fuel flow rate was not. Therefore, during the initial run at time zero, the combustion process is different to that at the design point, thereby altering the inlet temperature to the turbines. This changes the starting position of the trajectories on the turbine maps, because the inlet temperature is used to calculate the corrected speed N_C. The corrected speed is in-turn used to determine the mass flow rate from maps, which is also corrected using this same inlet temperature. Thus, arrows have been superimposed over the trajectories to indicate the their direction. With the aid of these arrows, it may be observed how the turbines decelerate and then accelerate during the slam acceleration.

The high pressure turbine, Figure 5.29, exhibits a better correlation than the low pressure turbine, Figure 5.29. The low pressure turbines accuracy deteriorates at lower speeds and mass flow rates. This is attributed to differences between the combustion model developed in this work and that of GSP.

Figures 5.31 to 5.35, illustrate the response of different engine parameters against time. Figure 5.31, illustrates the response of the pressure changes in the engine at different engine stations (please refer to Figure 5.3). These results are representative of the volume dynamics, which is based on the rate of change in pressures. As is evident, the correlation between the two models is very close. Each graph also provides the error at the end of the simulation for each parameter. In general, the errors in pressure do not

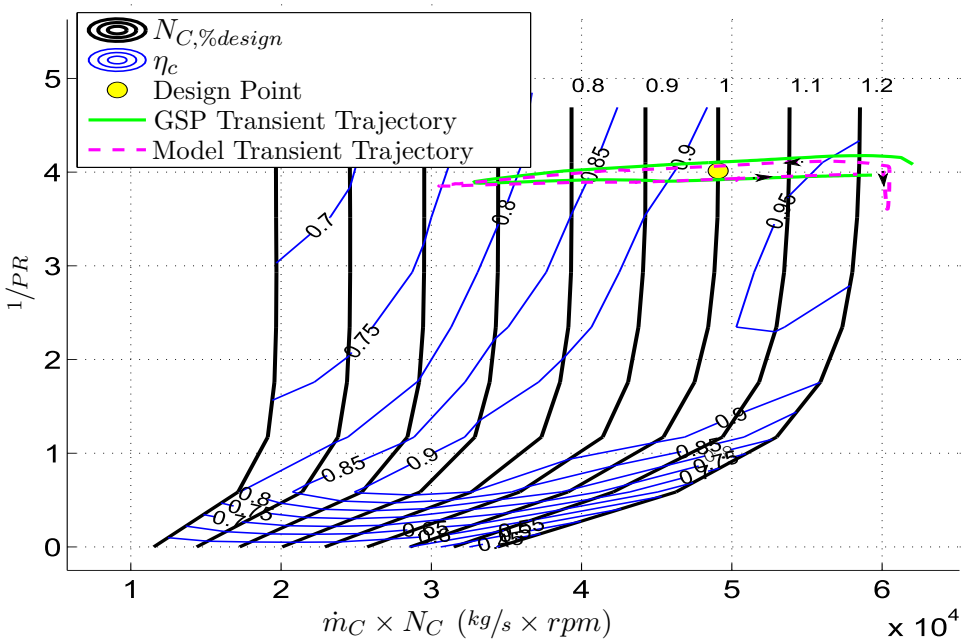


Figure 5.29: Engine model transient validation at static sea-level conditions: High pressure turbine map trajectories

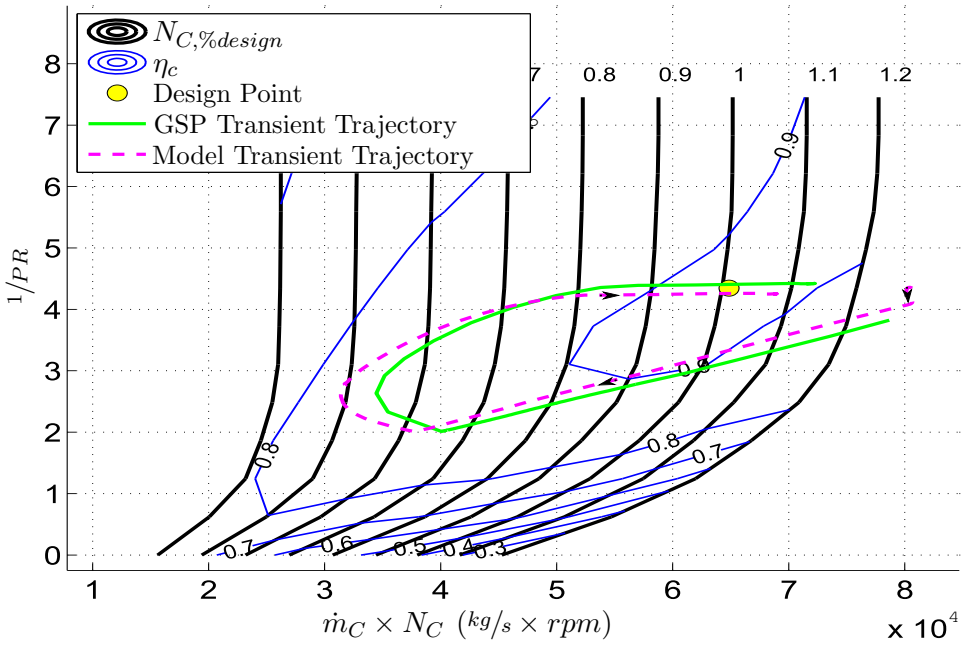


Figure 5.30: Engine model transient validation at static sea-level conditions: Low pressure turbine map trajectories

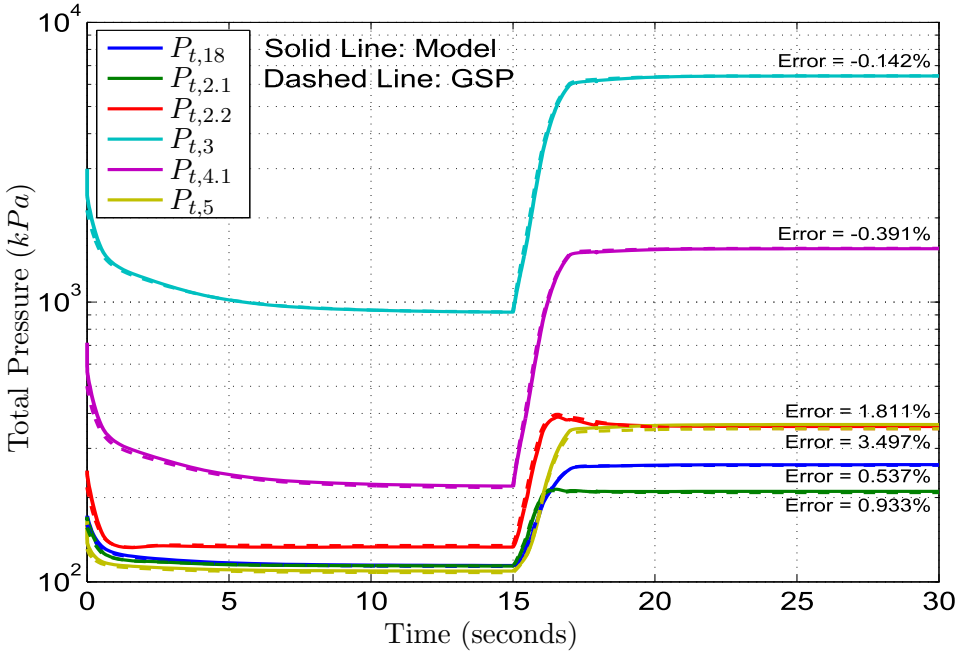


Figure 5.31: Engine model transient validation at static sea-level conditions: Pressure response curves

exceed 1%, except for at stations 2.2 and 5. Still, these errors fall well within 5%, and are considered to be sufficiently accurate.

Figure 5.32, illustrates the temperature responses at all the stations preceding the combustion chamber, or "cold end". Again, the correlations are close, and the error seldom exceeds more than 1%.

Figure 5.33, illustrates the temperature responses of the "hot end" (i.e. all stations located downstream of the combustion process). These responses clearly demonstrate the higher order dynamics that are inherent in an acceleration. This is evident from the camel hump, which occurs as a result of changing fuel-air ratio. The fuel-air ratio varies in this way, because of the dynamics in the mass flow rate through the engine. There is a fair correlation between the two models during these higher order effects. Furthermore, the final error in temperature does not exceed 3%.

Figure 5.34, illustrates the response of the spools during the acceleration, thus representing the rotor dynamics. As is evident, these dynamics are closer to first-order than the aero-thermodynamic transients. There certainly appears to be a close correlation in the most dynamic portion of the response, which deteriorates towards the end of the simulation to settle on the steady-state error. Thus the error is assumed to be a result of steady-state, and not dynamic, factors. Nonetheless, the low pressure spools erros are well within 1%, and the high pressure spool provides sufficient accuracy within 4% of GSP's results.

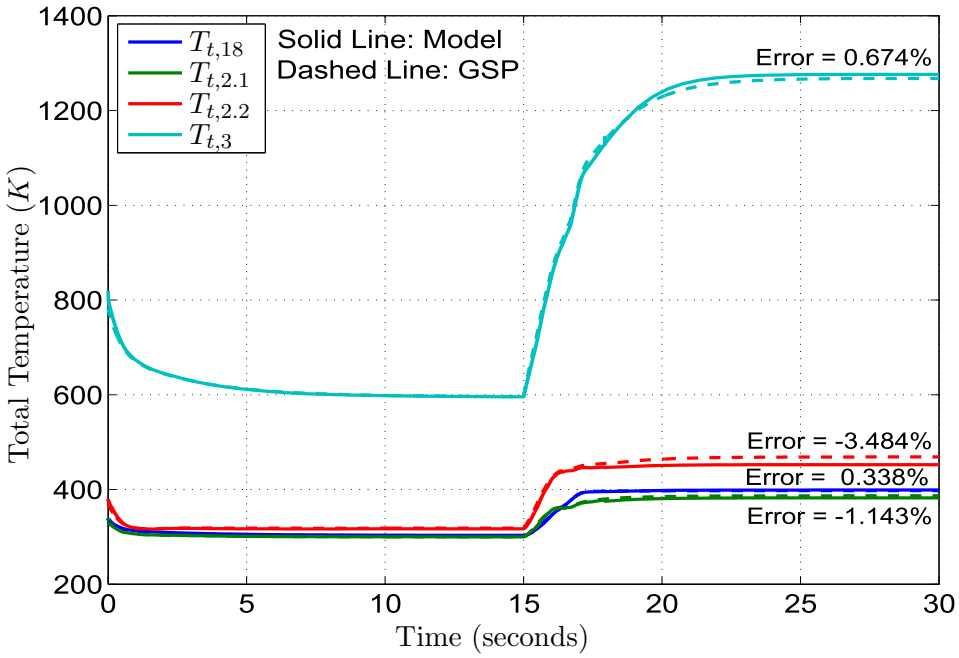


Figure 5.32: Engine model transient validation at static sea-level conditions: Cold end temperature response curves

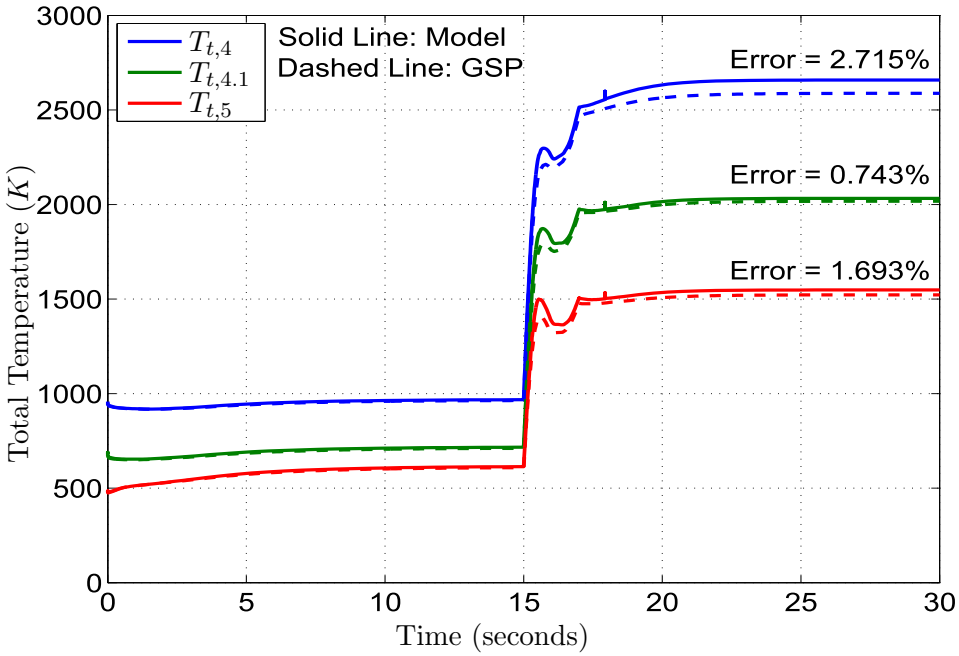


Figure 5.33: Engine model transient validation at static sea-level conditions: Hot end temperature response curves

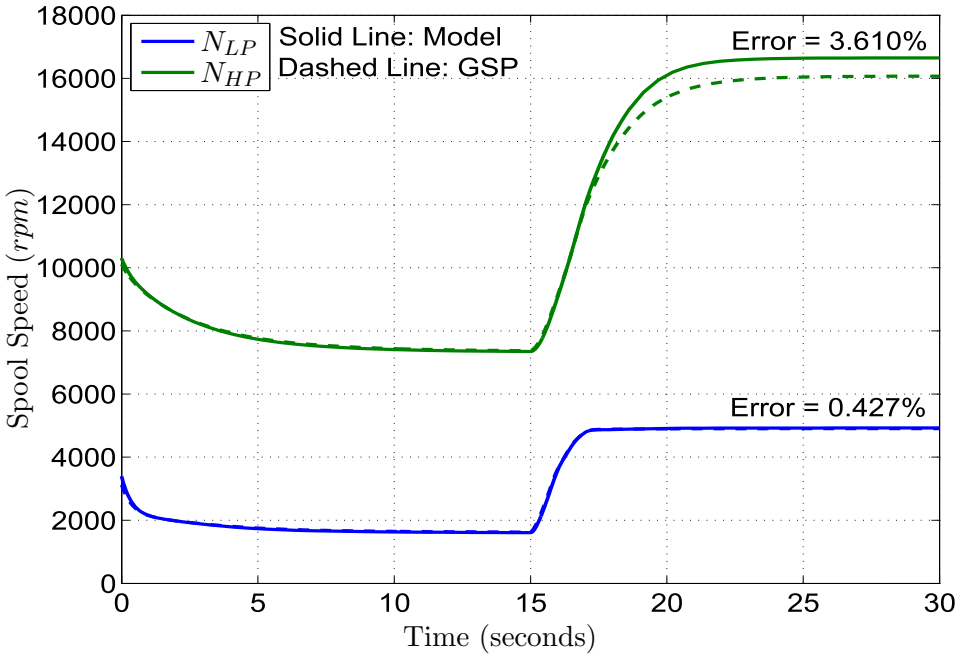


Figure 5.34: Engine model transient validation at static sea-level conditions: Spool speed response curves

However, the most important result, in the context of this work, is the thrust response to changes in fuel flow, which has been illustrated in Figure 5.35. This graph illustrates the response of ram drag, thrust produced at both nozzles and the overall net thrust produced by the engine. As is evident, there is an extremely close correlation between the two models during the dynamic portion of the response. The final error, as was the case with the spool response, is due to steady-state factors. Nonetheless, the errors are not very large, and in fact the net thrust (which is the most important) falls within 1.5% of GSP's final result. It is Figure 5.35, in particular, which proves the model developed in this work to be valid (at static sea-level conditions) for accurately predicting engine performance for varying fuel commands.

Cruise

Thus far, the engine model, developed in this work, has been validated. It remains to check if the model is still valid for transient operation at cruise conditions. This is therefore the most important validation test, as it is under these conditions that the the model will be used to determine engine performance during formation flight.

The same approach as before has been used to validate the engine model at these conditions. The model was initialised to its steady state operating

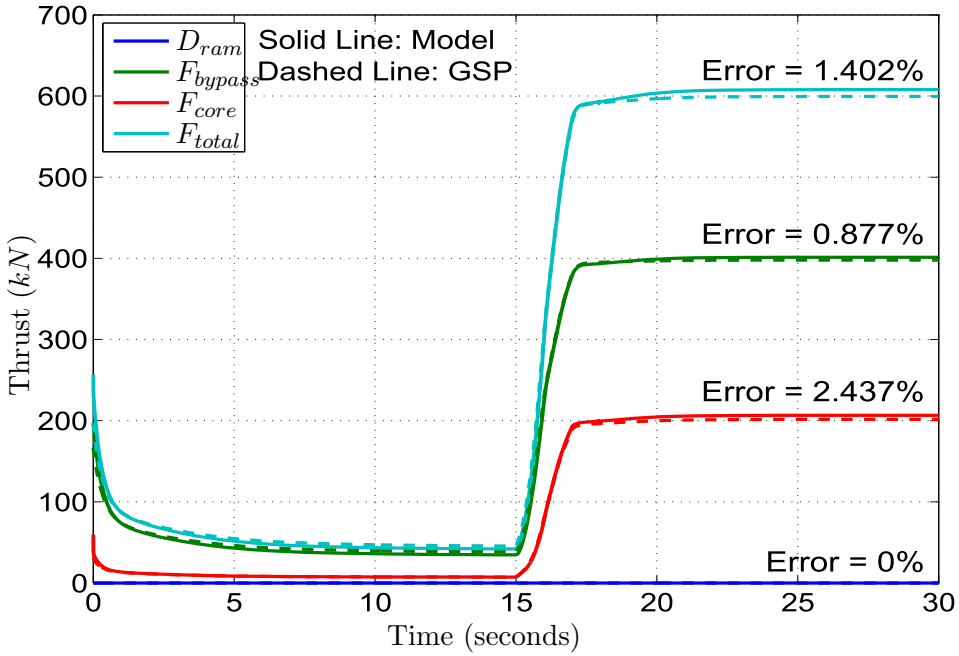


Figure 5.35: Engine model transient validation at static sea-level conditions: Thrust response curves

point, correlating to the lowest allowable fuel flow rate, 0.05 kg/s. The model was allowed 5 seconds to stabilize, should there be any imbalances still present. Thereafter, the fuel was increased linearly to a maximum value of 2.815 kg/s. The shortest permissible time for this fuel flow increase was 20 seconds. This is because the model is far more sensitive to changes in fuel flow at these conditions. Thereafter, the simulation was run until a minute had passed.

Figures 5.36 to 5.39, illustrate the transient trajectories of each compressor, which indicate a close correlation between the two models in general. The only significant errors, are those of the intermediate pressure compressor, Figure 5.38. The possible cause of these errors has already been discussed in the previous validation section, and is therefore not repeated here. It is noted, however, that these errors do not detract significantly from the overall model's accuracy.

Figures 5.40 and 5.41, illustrate the correlation between the two model's trajectories on the high and low pressure turbine maps, respectively. Again, there is a fair correlation in both figures, but with the low pressure turbine exhibiting significant discrepancies. However, these discrepancies do not appear to effect the overall engine model results.

Figure 5.42, illustrates the engine model's pressure responses. As is evident, there is an excellent correlation between the two models, with the final steady state error less than 1% for all stations.

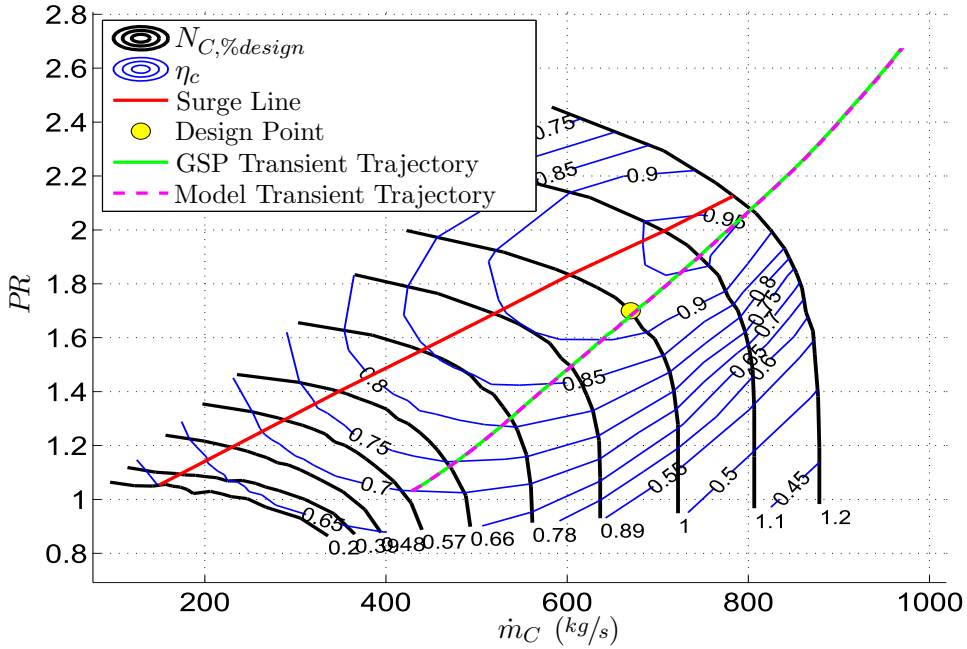


Figure 5.36: Engine model transient validation at cruise conditions: Low pressure (bypass) compressor map trajectories

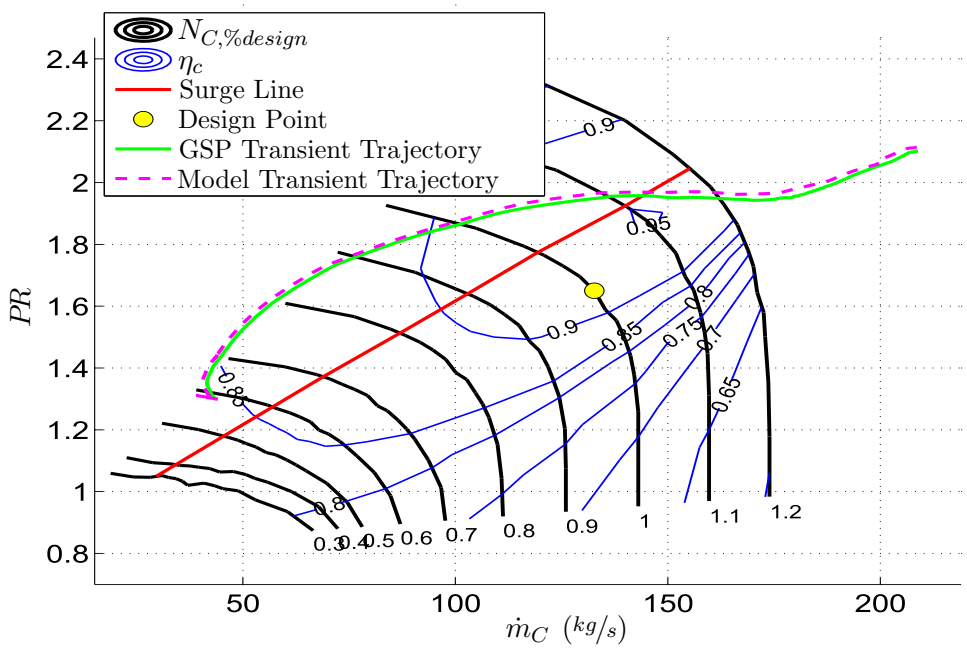


Figure 5.37: Engine model transient validation at cruise conditions: Low pressure (core) compressor map trajectories

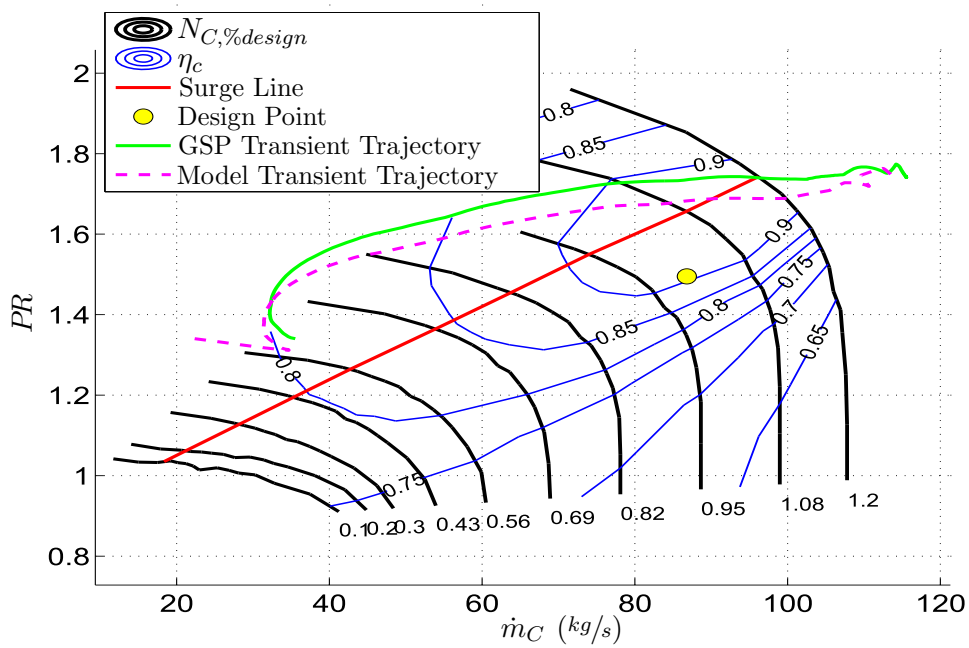


Figure 5.38: Engine model transient validation at cruise conditions: Intermediate pressure compressor map trajectories

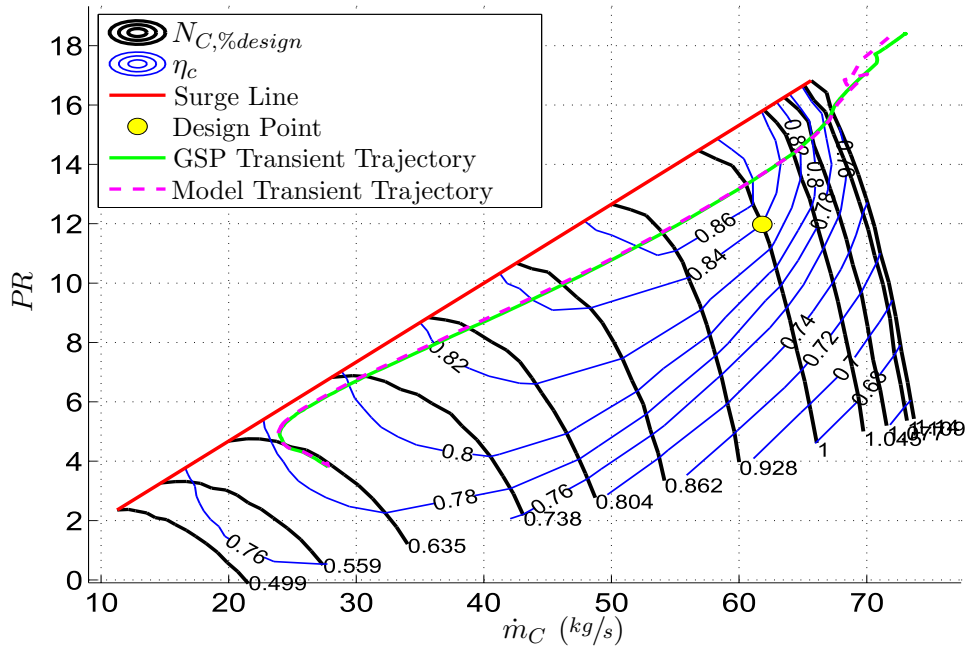


Figure 5.39: Engine model transient validation at cruise conditions: High pressure compressor map trajectories

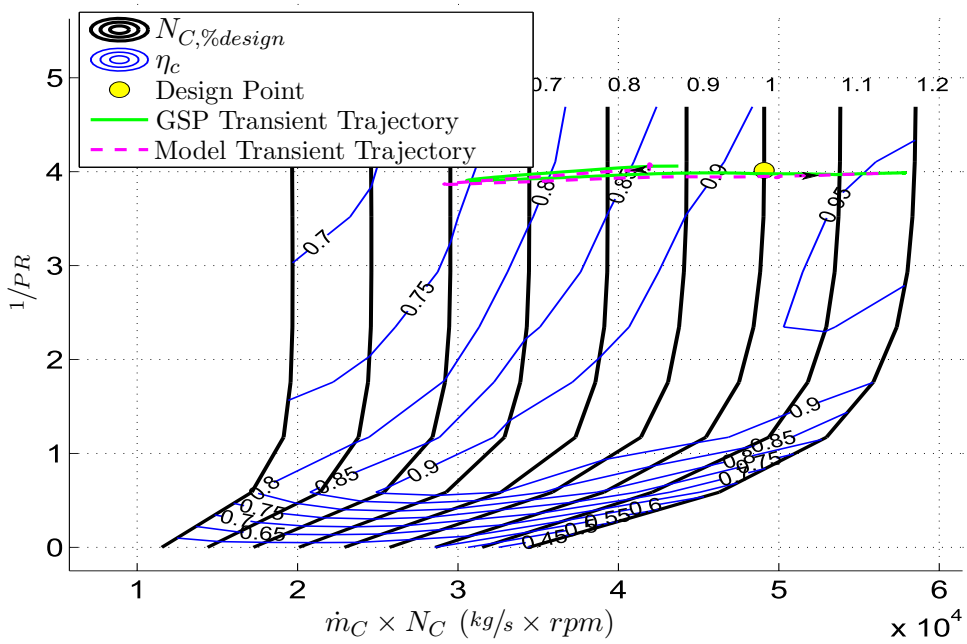


Figure 5.40: Engine model transient validation at cruise conditions: High pressure turbine map trajectories

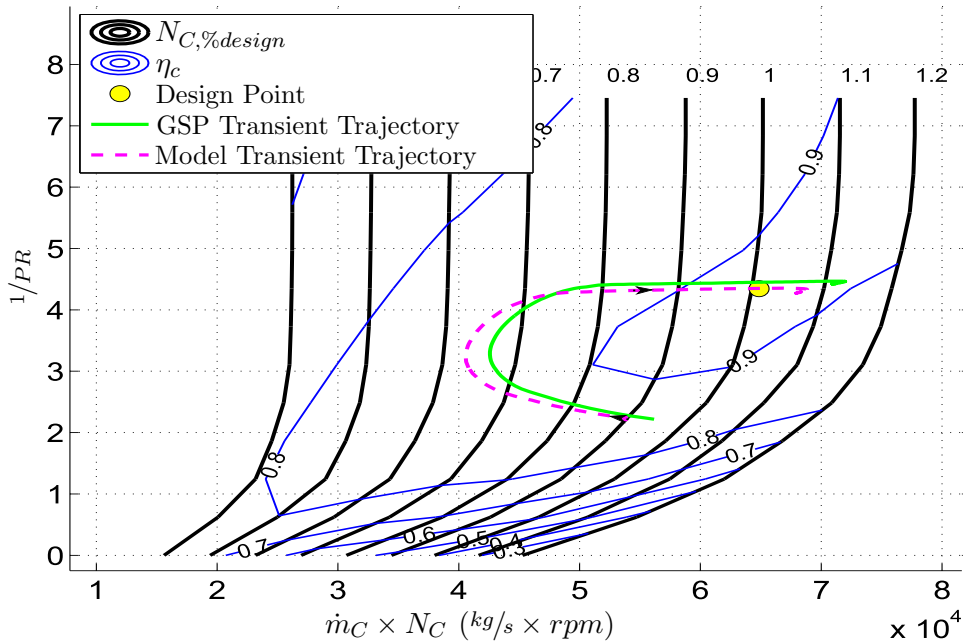


Figure 5.41: Engine model transient validation at cruise conditions: Low pressure turbine map trajectories

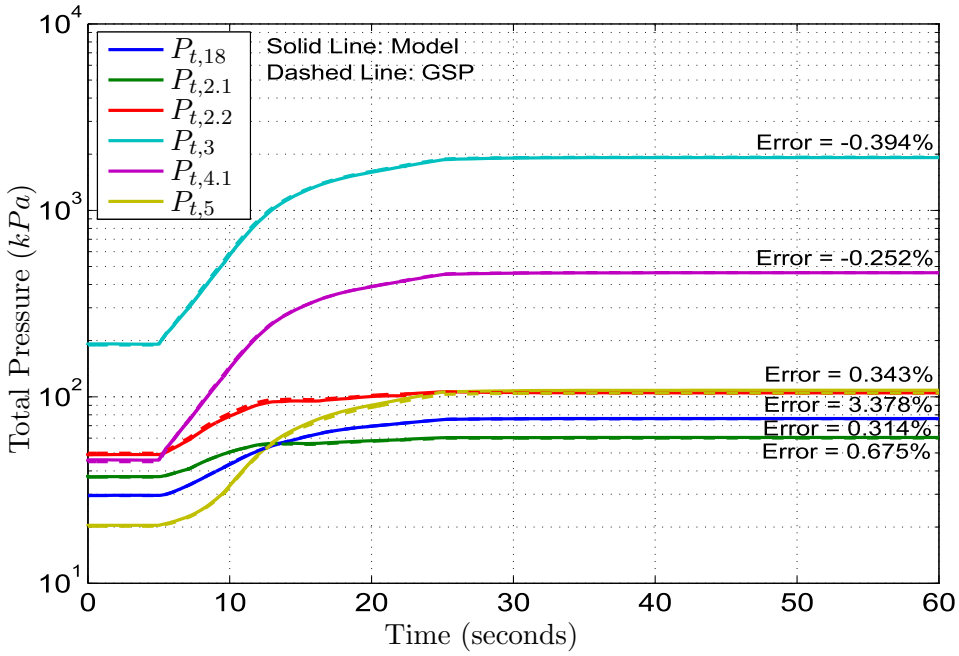


Figure 5.42: Engine model transient validation at cruise conditions: Pressure response curves

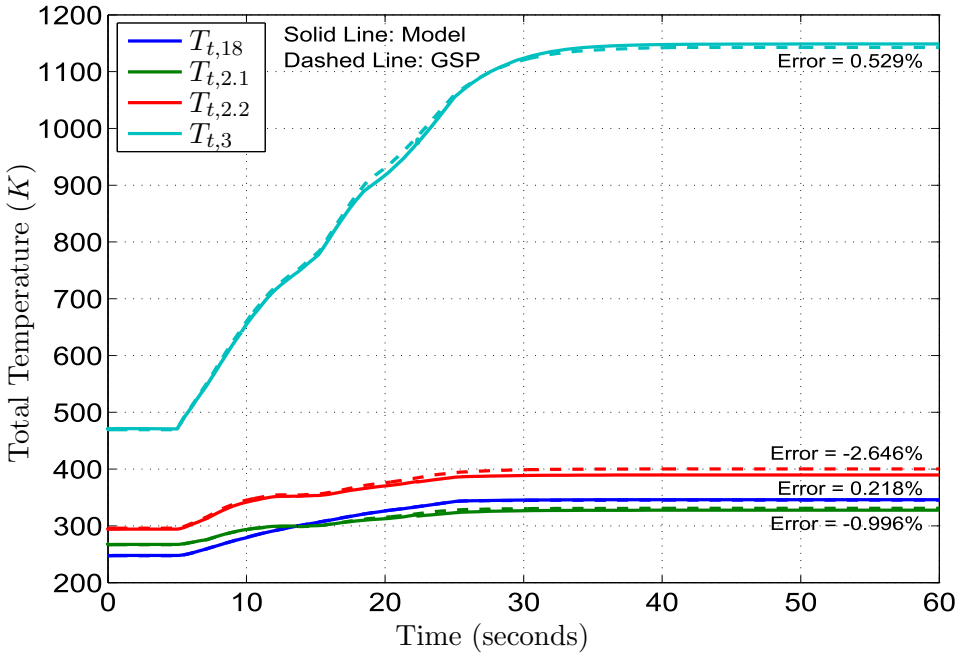


Figure 5.43: Engine model transient validation at cruise conditions: Cold end temperature response curves

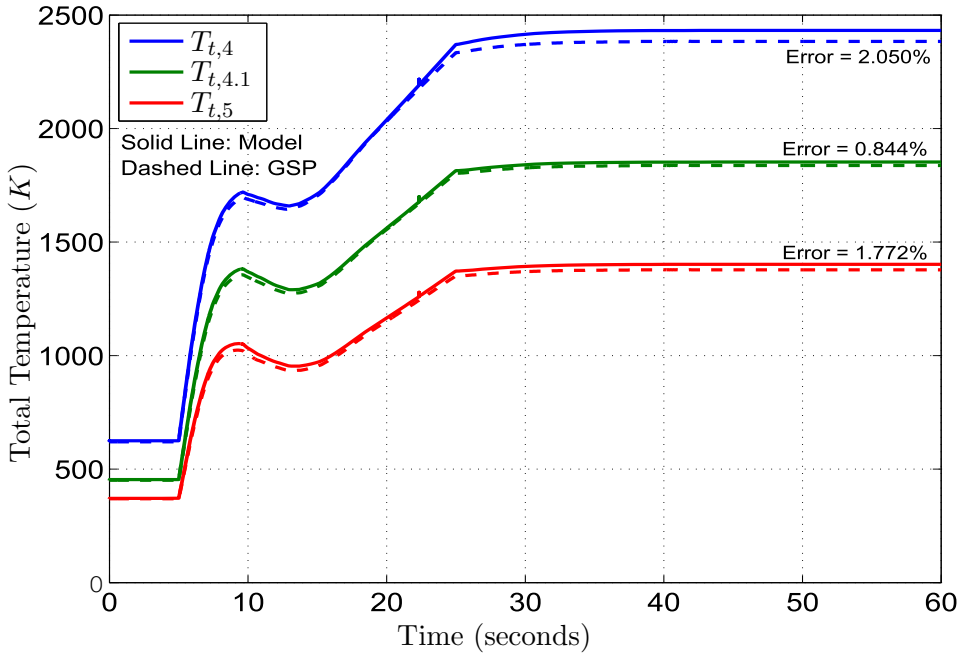


Figure 5.44: Engine model transient validation at cruise conditions: Hot end temperature response curves

In addition, there is also an excellent accuracy in the temperature responses of the the cold end of the engine, Figure 5.43. The Temperature response accuracy is almost maintained in the hot end of the engine, Figure 5.44, with errors barely exceeding 2%. Figure 5.44, also illustrates very nicely the non-linear effects that exist in the temperature responses. Furthermore, there is excellent correlation between the two models over these regions of higher order.

Once again, the spool speed response, Figure 5.45, are found to be in close agreement with one another. The high pressure spool exhibits a slightly higher final steady-state error, but this does not exceed 3%

Finally, the thrust responses are depicted in Figure 5.46. Again, there is excellent correlation between the two models, with the error not exceeding more than 2%. In fact, the net thrust produced is found to be within 1.5% of that predicted by GSP's model.

Therefore, the engine mode developed in this work, is deemed valid for predicting transient performance at cruise conditions.

5.8 Limitations

The engine model described thus far does have many limitations in its capabilities. This section discusses those limitations, as well as their possible implications with respect to the results produced in this work.

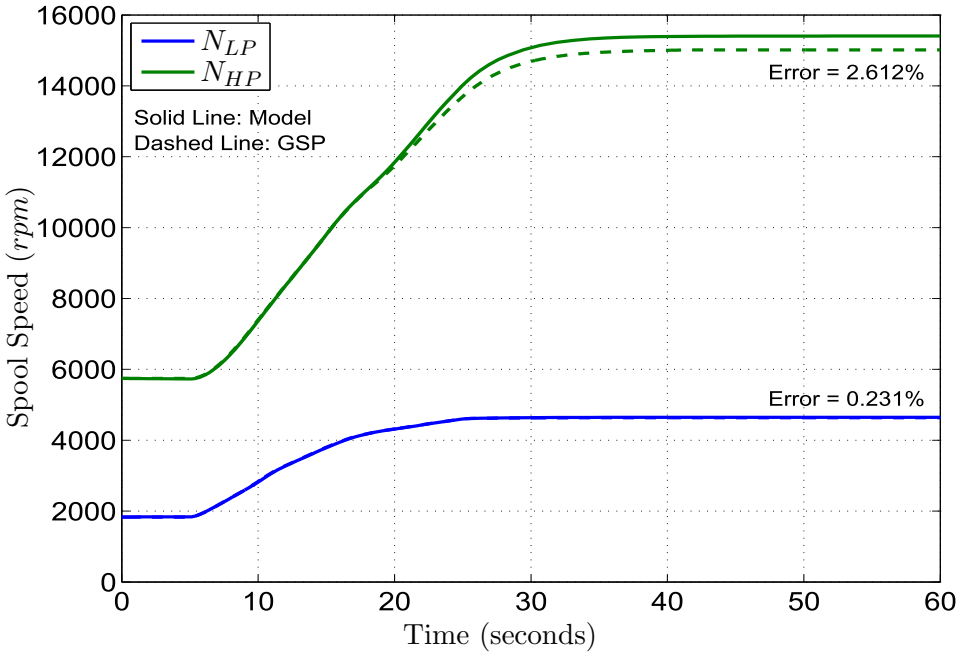


Figure 5.45: Engine model transient validation at cruise conditions: Spool speed response curves

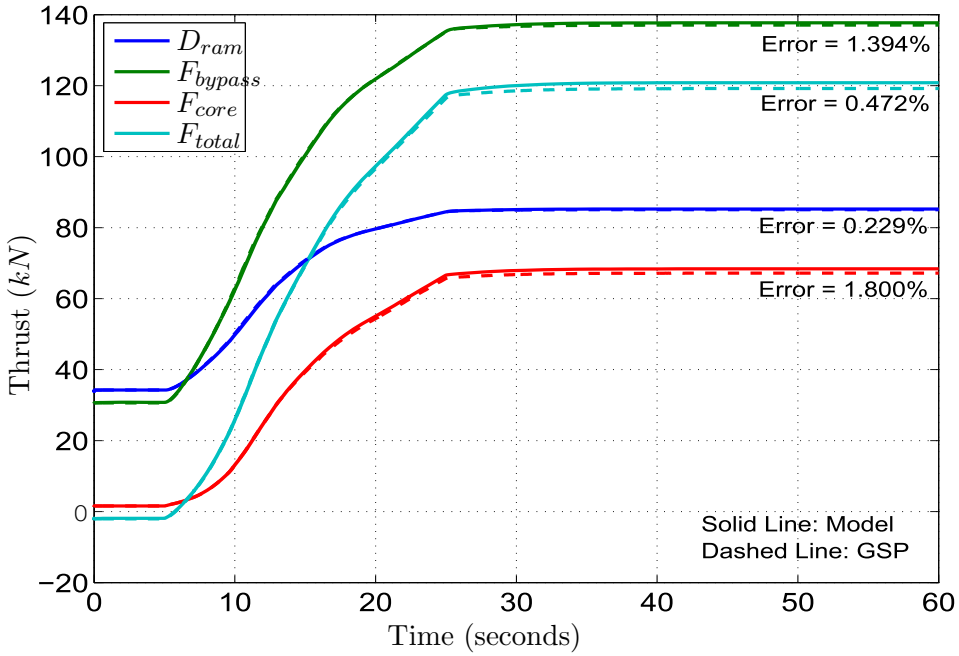


Figure 5.46: Engine model transient validation at cruise conditions: Thrust response curves

5.8.1 Fan Model

It is first important to discuss any potential issues with the component models developed in this work. The first significant limitation, is the simulation of the fan model in off-design and transient conditions. Unfortunately the model developed in this work does not allow for radial flow to take place between the two parallel compressors representing the fan. This assumption causes the core map to operate a lot closer to, and often breech, its surge margin. For example, considering a sudden acceleration in a more realistic scenario, the pressure ratio across the core compressor would increase drastically before the spool has time to react. As a result, the operating point moves up the speed isoline, thereby decreasing its mass flow rate and approaching the surge line. However, not forgetting that the fan is in reality one compressor, flow moves radially from the bypass map into the core part of the engine, thus effectively increasing the flow through the core, and thereby reducing the onset of surge. Obviously this shift of flow moves the bypass' operating point closer to its own surge margin, but only partially. Therefore, the "surge load" is shared by the two parallel compressor maps. This has obvious implications on the accuracy of the engine performance predicted, because it not only effects overall fan efficiency, but also the mass flow rate through the entire engine core. However, at the current stage and scope of this work, any accuracy lost is not deemed significant. Nonetheless, it is strongly encouraged that a higher fidelity map model be developed for further investigations. Particularly, concerns regarding surge, need to be accounted for more comprehensively in future work.

5.8.2 Combustion Model

Another component that includes limitations, is the combustion model. Combustion is the heart of the engine's performance, and therefore it is very important that it be modelled as accurately as practically possible. The combustion model in this work ignored the effects of dissociation, thereby in turn ignoring many possible species of combustion products. Throughout the validation of hot end components, the effects of the discrepancy between the combustion models (of this work and that of GSP) were clearly evident. In addition, the inclusion of dissociation effects will aid the prediction of the engine emissions, which may become more important as this topic progresses. Nonetheless, for the purposes of this initial study, the fidelity of the combustion model developed is deemed to be adequate.

5.8.3 Compressor Bleeds

This model, currently, does not have the capabilities to model compressor bleed systems. Compressor bleeds are often used for auxiliary aircraft systems, and may have a significant effect on the operation of the engine. In

addition, bleeds may used to pass flow to other parts of the engine to improve the overall engine's performance. An example of this is turbine blade cooling. In the former scenario, information on bleed systems is very difficult to obtain, and seems to be, for now, an unnecessary complication to the current engine model in the context of this work. In the latter scenario, the engine model would require heat soakage capabilities, to predict the effect on turbine cooling. Heat soakage is another complex addition to the current engine model and is discussed next.

5.8.4 Heat Transfer and Soakage

The transfer of heat between the gases within the engine, the engine parts and engine surroundings does effect the performance of the engine. However, it is fairly complex to incorporate within an engine model. The inclusion of heat soakage is not deemed necessary for this particular work, because it limits the practicality in including the engine model within the aircraft models, within a formation model. As the engine model's fidelity increases, its complexity increase, thus reducing its computational efficiency. For the purposes of this work, the engine model is required to be of a high computational efficiency to be practically included within the formation model. The improvement in accuracy achieved through the inclusion of heat soak age, does not warrant the loss in computational efficiency. Therefore, the exclusion of heat soakage is not deemed to impact significantly on the results produced by this work.

5.8.5 Steady-State Model

As has been mentioned previously, the model developed is designed for transient simulations. However, it would be useful to include a mode that is solely dedicated to steady-state calculations. This has no implications on the results obtained from this work. However, it would provide a convenient tool for initialising the transient model.

5.8.6 Engine Control Unit

The final, yet most influential limitation, is that of the engine control unit. A very crude control unit was included; one that limits the maximum and minimum fuel rates as well as the rising and falling slew rates. This crude fuel control unit may result in engine response times which are unrealistic, which will in-turn effect formation dynamics and, therefore, the predictions on fuel savings achieved by the formation. A more sophisticated control system, will constantly monitor the the engines operating parameters to augment the commanded fuel flow rate such that the engine operates safely and at its highest efficiency. This could significantly effect the results produced in this work.

5.9 Summary

This chapter has presented the engine model developed to predict engine performance during dynamic throttling in formations flight. An aerothermodynamic approach was used in modelling the engine, and was based on the Inter-Component Volume method. This model also adopted the twin-map approach for modelling the fan's performance, as well as the inclusion of variable specific heats to improve fidelity.

The components were modelled thermodynamically, and have been extensively validated in isolation, which has been detailed in Appendix E. The rotary components (turbomachines) are based on steady-state characteristic maps, the handling of which has been described and validated in Appendix D.

The engine model, as a whole, has been described with reference to its dynamics (both rotary and volume). The design point calculation and transient initialisation of the engine model has also been described. Most importantly, the model has been validated against GSP commercial software. The model was used to simulate General Electrics CF6-80C engine. Validation tests were conducted for steady-state and transient operation at static sea-level and cruise conditions. There was close agreement between both models, and therefore the model developed in this work was proven to be valid.

A simple fuel control system has been incorporated within the model. This control system limits the maximum and minimum fuel flow rates, and rising and falling slew rate. The limitations of the model have also been discussed. Nonetheless, the model developed is deemed to be suitable and sufficiently accurate for tentative investigations into the fuel savings achievable in formation flight.

Chapter 6

Combined Formation and Engine Model

The aerodynamic theory presented thus far, was applied to a twin aircraft formation. The aircraft in the formation, are identical and based on Boeing 747-100 data, who's details are presented in Appendix B. The transient engine model was incorporated within the formation's dynamic simulation, to provide an accurate evaluation of the effects of inefficient engine transients on fuel savings. This chapter first examines the performance of the formation were it flying in still conditions, as a point of reference. The subsequent sections investigate the effects of atmospheric turbulence on the formation's performance.

6.1 Formation Benefits in a Still Atmosphere

The formation was first modelled in the absence of atmospheric turbulence, and therefore in steady-state conditions. Here, the longitudinal trim conditions were calculated for both the lead and trail aircraft, at various vertical and lateral separations. The results for each aircraft were compared against one another, with the leader representing an aircraft flying in isolation. Because of parasitic drag, the induced drag savings on the trail aircraft are not necessarily indicative of fuel savings, even during steady-state conditions. The actual drag savings are also a result of the trimmed condition on the trail aircraft, which includes changes in not only throttle, but incidence and elevator deflections. In addition, the thrust line is not parallel to the drag force, or even the longitudinal body axis, and does not pass through the aircraft's c.g.. Therefore, the thrust contributes to the aircraft's pithing moment, thereby effecting the trim incidence and elevator deflections. Finally, the trimmed thrust setting is found at some off-design engine operating point, at an engine efficiency different to that of the isolated aircraft's engines. Contour plots of the follower's trimmed drag, thrust and fuel savings

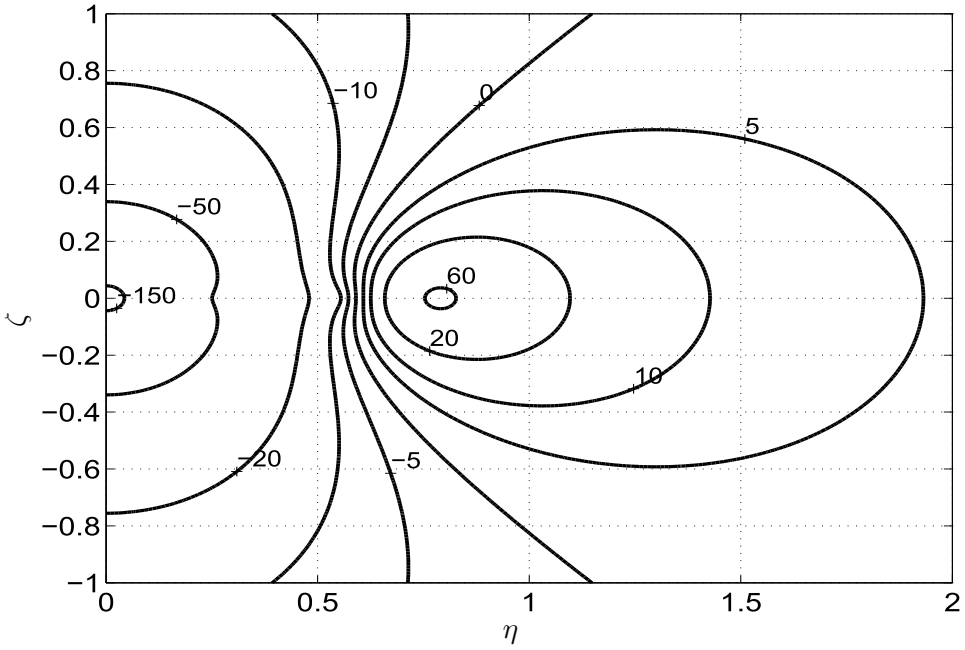


Figure 6.1: Follower's drag savings at trimmed, steady-state conditions: $\left(1 - \frac{\Delta D_F}{D_I}\right) \times 100$, (%)

at various lateral and vertical separations, are depicted in Figures 6.1 to 6.3 respectively.

Figure 6.4 views these trimmed parameters from a more illustrative perspective, at $\zeta = 0$. From this figure, the differences between the performance parameters are more clearly visible. The drag and thrust savings are almost exactly the same, due to the small angle between the thrust and drag forces. However, there are significant differences between the fuel saving and that of the thrust and drag savings. This, as has already been mentioned, is attributed to variations in engine efficiency as it operates further away from its design point, further illustrated by the TSFC curve in Figure 6.4. Here, it is clearly visible how the engine's operational efficiency decreases at the two extreme aerodynamic points (i.e. trail directly behind the leader, and at the point of greatest drag saving).

From Fig. 6.4, the best saving is located at a spanwise separation of $\eta \approx 0.79$ and $\zeta = 0$ where the drag saving, including parasitic drag, is approximately 73.5%, which corresponds to a fuel saving of approximately 64.8%. However, this is perhaps not the most practical region in which to fly. This is for a number of reasons, the first being that the gradients of the curves are steepest at this region, and in fact change sign. Therefore, aircraft control becomes more difficult, due to amplified aircraft dynamics in perturbed flight. The second reason is that the engine is operating at

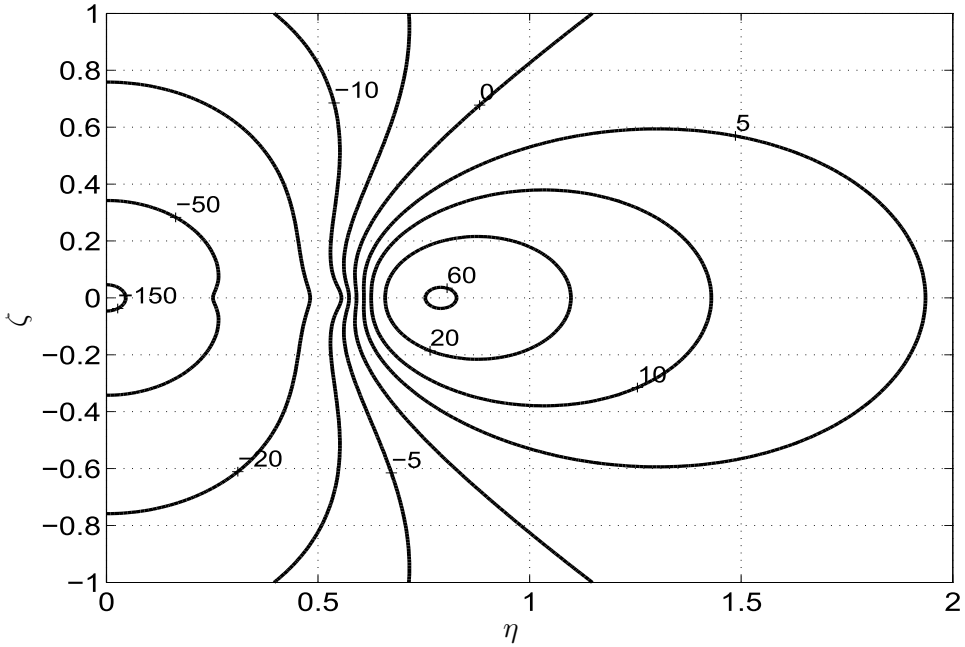


Figure 6.2: Follower's thrust savings at trimmed, steady-state conditions: $\left(1 - \frac{\Delta F_{th,F}}{F_{th,I}}\right) \times 100$, (%)

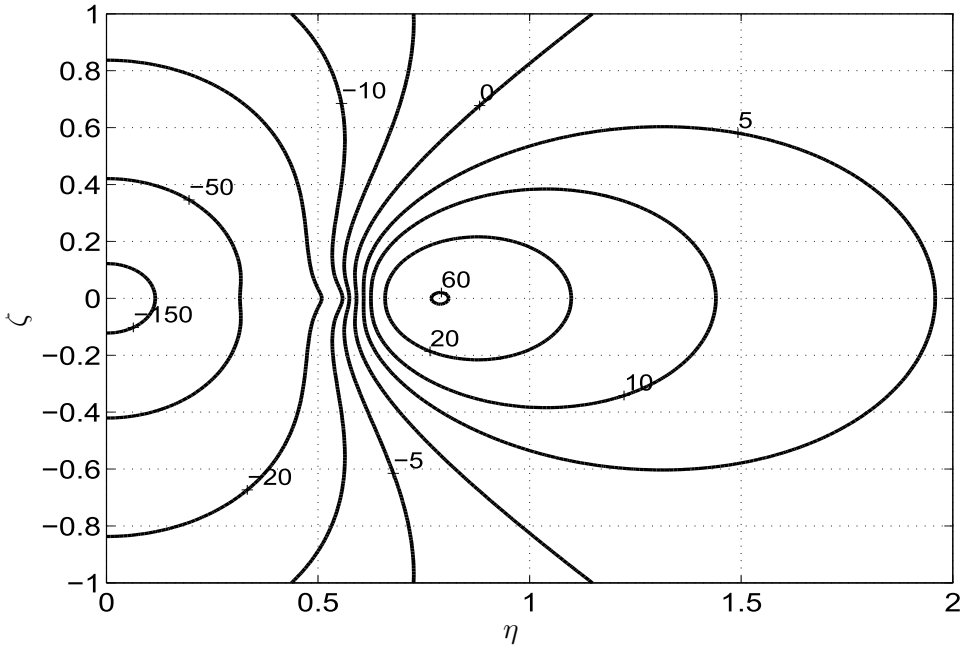


Figure 6.3: Follower's fuel savings at trimmed, steady-state conditions: $\left(1 - \frac{\Delta \dot{m}_{f,F}}{\dot{m}_{f,I}}\right) \times 100$, (%)

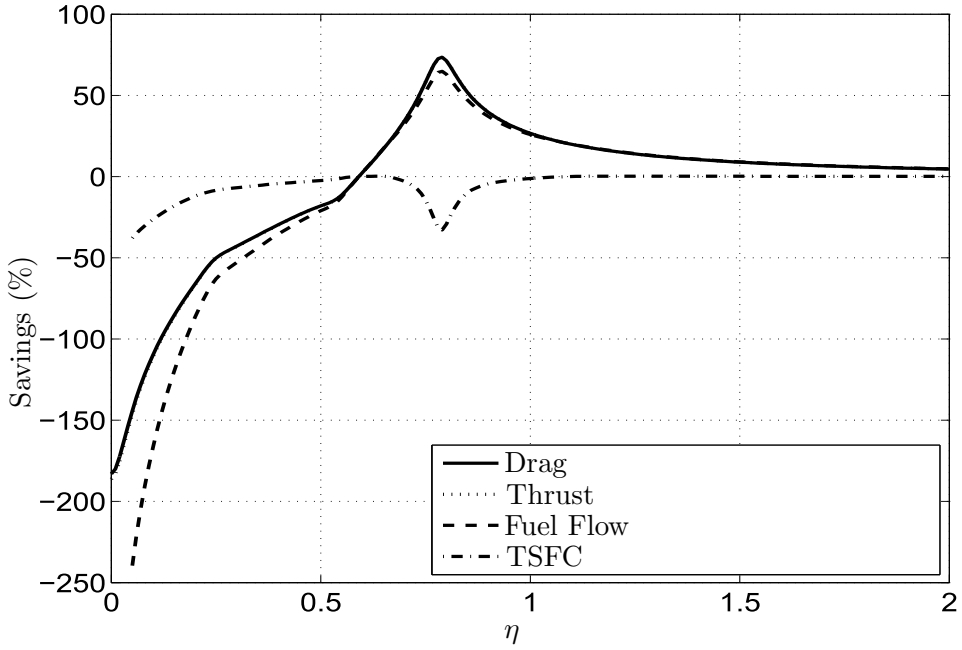


Figure 6.4: Follower's drag, thrust and fuel savings at trimmed, steady-state conditions, with $\zeta = 0$

its lowest efficiency at this point (which is evident from the TSFC curve, and not considering the point directly behind the leader), and therefore its transient perturbations about this point are likely to result in an even lower engine operational efficiencies. It is reasonable to assume that engine operational efficiency during transients will be less affected when accelerating (or decelerating) between steady-state points of equivalent TSFC. In light of the above discussion, it would be better for the trail aircraft to fly further outboard of the leader, where gradients are gentler and engine TSFC is closer to a constant value. From Fig. 6.4, such a region is located outside of $\eta = 1$. However, it is not desirable to move too far outboard as the formation benefits diminish quickly. A promising region may be located between $\eta = 1$ and $\eta = 1.5$.

6.2 Appropriate Separations in Atmospheric Turbulence

As already illustrated, there are large zones where the trailing aircraft may receive aerodynamic benefit. Recalling that in this work the lateral and vertical aircraft motions are assumed to be controlled perfectly, and therefore the vertical and lateral separations between the aircraft are fixed. It now remains to be decided at what fixed lateral and vertical separation the

trail aircraft could fly practically, whilst optimising benefit. To do this, the possible effects of atmospheric turbulence need to be taken into account. In this work it has been assumed that the trailing vortices of the leader will shift about with the atmosphere. This results in a change in the effective separation (Equations 3.44 and 3.45) between the trailing aircraft and the leader's wake. Subsequently, expressions that allow for the determination of the wake shifted region, as experienced by the trailing aircraft, may be obtained:

$$\eta_{eff,r} =\approx \xi \left(\frac{v_g}{V_\infty} \right) \tag{6.1}$$

$$\zeta_{eff,r} =\approx \xi \left(\frac{v_g}{V_\infty} \right) \tag{6.2}$$

To determine the "buffer region" of the shifted wake, all that is required is the nominal stream-wise separation between aircraft, the speed of the formation and the turbulent gusts in the vertical and lateral directions. Ning *et al.* [13], recommend a nominal stream-wise extended separation of 20 spans, where the trailing vortices have rolled-up completely, and the effects of vortex decay are negligible. The turbulent gust velocities are obtained from the turbulence model based on the formation's speed and altitude at cruise. Subsequently, the vortex's deviation from its expected position (i.e. $(\eta_{eff,r} - \eta)$ and $(\zeta_{eff,r} - \zeta)$) is recorded by running the turbulence model for a sufficient amount of time to allow for a meaningful statistical result. The maximum deviations in the η and ζ coordinates may represented by an ellipse to estimated the buffer region in which the vortex shifts from its expected position (as seen by the trail aircraft). In light turbulence conditions this region is small enough to maintain a region of 60% fuel saving, and is is therefore not considered to affect the formations performance significantly. At moderate and severe turbulence levels, however, the buffer region is much larger, and are depicted in Figures 6.5 and 6.6.

Figures 6.5 and 6.6 illustrate the buffer regions for moderate and severe turbulence, respectively. Because the buffer region becomes larger with increased turbulence, it seems intuitive that the trailing aircraft must move outboard of the leader to fly in a more practical region. In this work, this practical region is estimated to be at the lateral separation where the vortices do not shift to a position of negative benefit. In so doing, the aircraft will not experience a change in sign of the induced forces and moments, which would adversely affect the controllability of the aircraft. It is estimated that this region may offer the best compromise between practicality and fuel savings. The lateral separation of best compromise was determined graphically, and was found to be around $\eta = 1$ and $\eta = 2$ for moderate and severe turbulence respectively, which is depicted in Figs. 6.5 and 6.6.

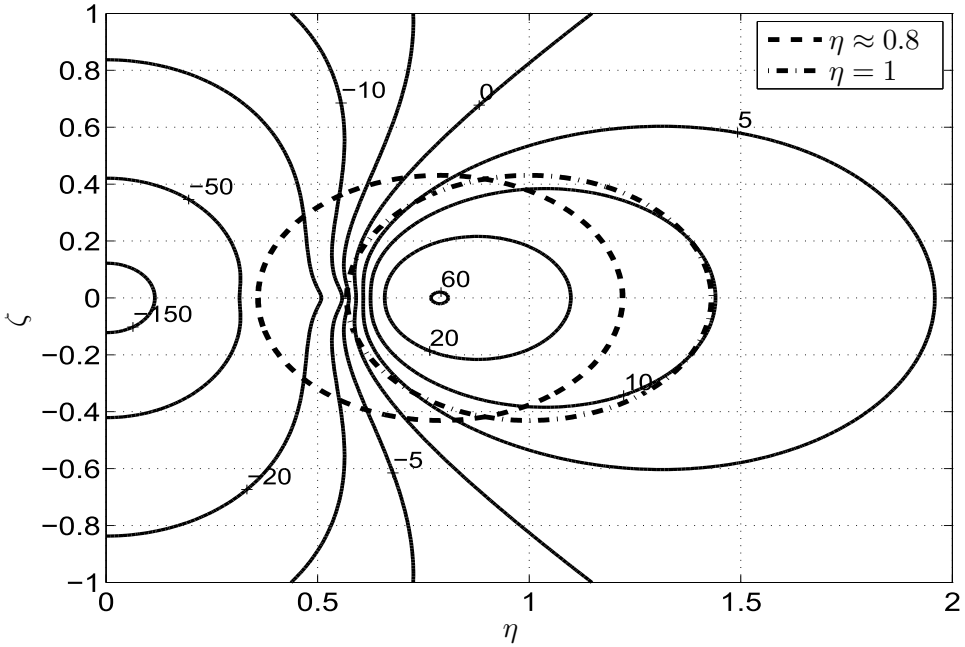


Figure 6.5: Vortex shift buffer regions, as seen by the trailing aircraft in moderate turbulence

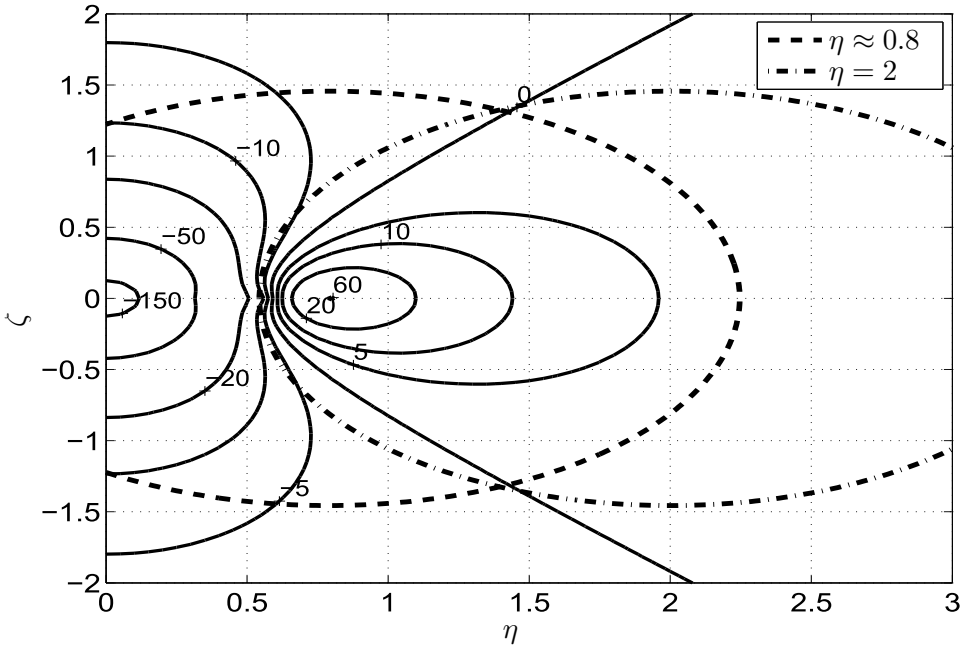


Figure 6.6: Vortex shift buffer regions, as seen by the trailing aircraft in severe turbulence

Because the effect of light turbulence on wake shifting is negligible, it is fair to assume that the effects of dynamic throttling will also be insignificant. On the other hand, severe turbulence is too extreme to be considered practical for formation flight, as is evident from Fig. 6.6 . Ning *et al.* [13], through a much higher fidelity study, also came to the conclusion that formation flight is not practical in severe turbulence. This was affirmed by Bizinos and Redelinghuys [25], based on their study on passenger comfort. Therefore, like Ning *et al.*, the remainder of this work will focus on the effects of moderate turbulence levels only.

6.3 Fuel Flow Schedules

Before the results of the various simulations are discussed, it is necessary to mention briefly how the fuel flow scheduling works. The proportional gain fuel controller, schedules the fuel demand according to the following equation:

$$\dot{m}_f = G\left(\xi - \xi_{\text{nominal}}\right) + \dot{m}_{f,\text{trim}} \tag{6.3}$$

where, G represents the gain value. The demanded fuel flow rate is, therefore, a function of the error in streamwise separation. It is noted that the demanded fuel flow rate is not the actual fuel flow rate that reaches the engine's combustion chamber. This is because the engine model also meters the demanded flow rate, so that the maximum and minimum allowable rates are not exceeded. In addition, the engine model also contains a slew rate limiter, which monitors the rate at which the fuel flow rate may be adjusted without causing the model to crash.

6.4 Fuel Savings in Moderate Turbulence

The formation was submersed in moderate turbulence and tested at two different lateral separations; i.e. the "sweet spot" at $\eta = 0.79$, and further outboard at $\eta = 1$, to examine the formation's dynamics at the conditions specified by Fig. 6.5. For each lateral separation, the formation's fuel consumption and station-keeping precision was evaluated for different control gains. The flight time for each test case was set to an hour, so as to obtain a meaningful statistical representation of the formation's performance. Figures 6.7 to 6.10 depict the formations performance when submersed in moderate turbulence.

Figure 6.7 illustrates the mean fuel saving of the trailing aircraft, for different levels of control gain. As expected, an increase in gain results in a decrease in fuel saving with an associated increase in mean position error and, as illustrated by Figs. 6.7 to 6.8. It is also evident, from Fig. 6.7, that the formation is more able to approach the benefits achievable in a still

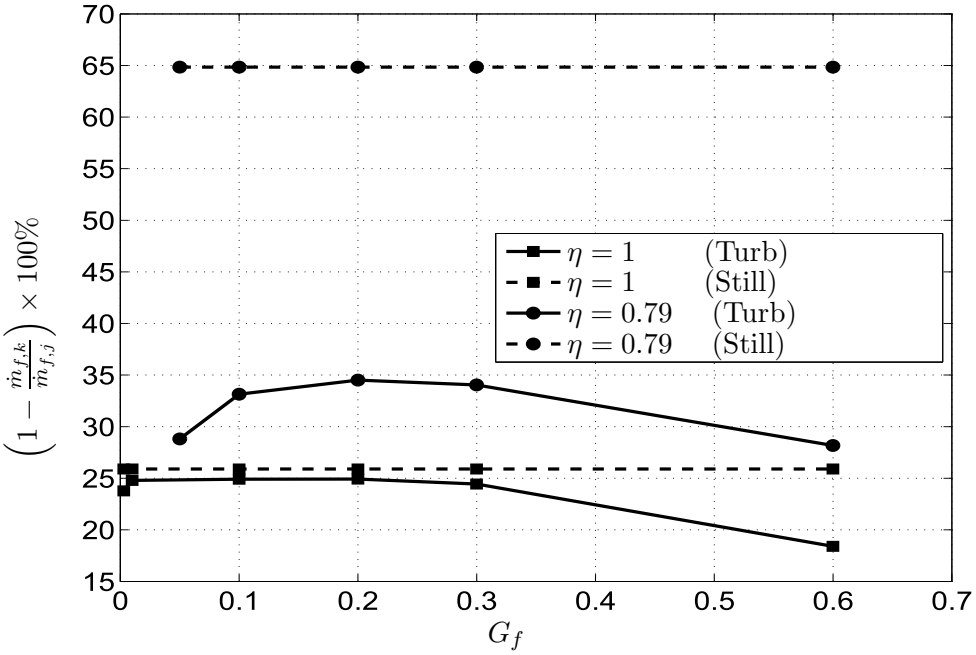


Figure 6.7: Results of mean fuel savings at different lateral separations and control gains, when submersed in moderate turbulence

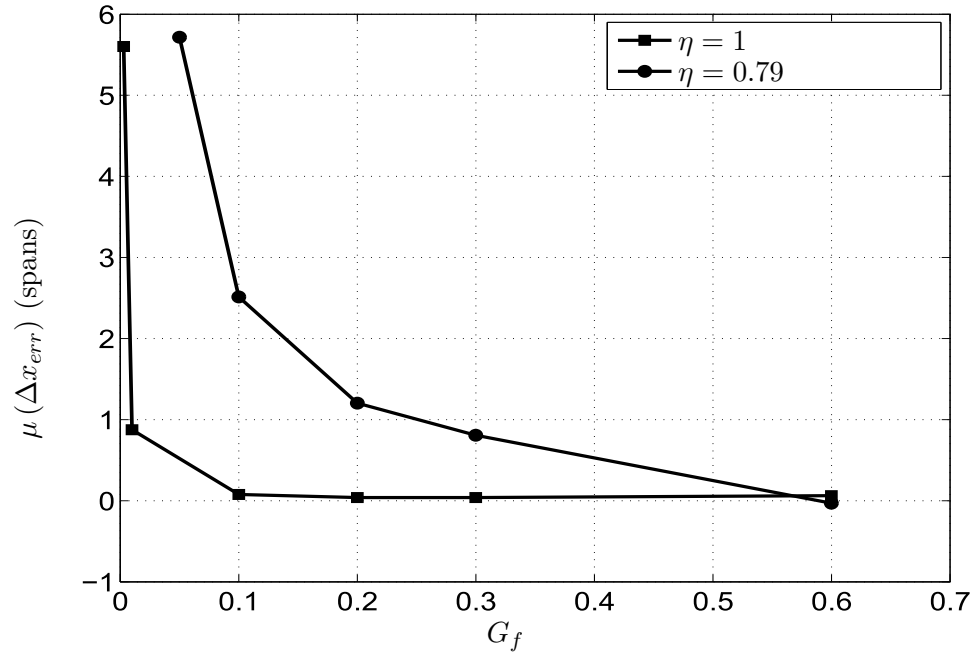


Figure 6.8: Results of mean separation error at different lateral separations and control gains, when submersed in moderate turbulence

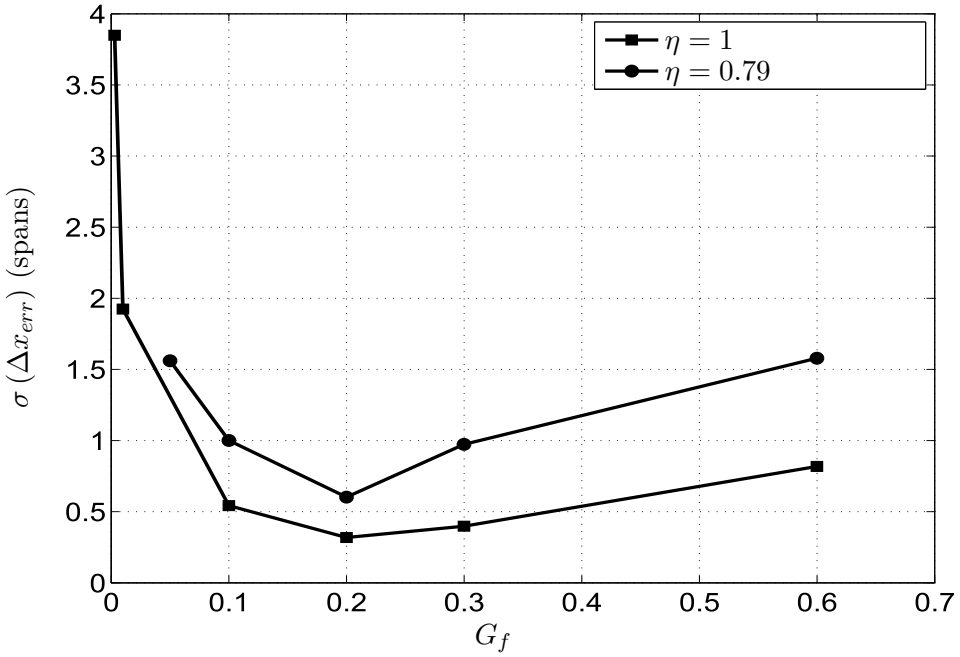


Figure 6.9: Results of separation error standard deviation at different lateral separations and control gains, when submersed in moderate turbulence

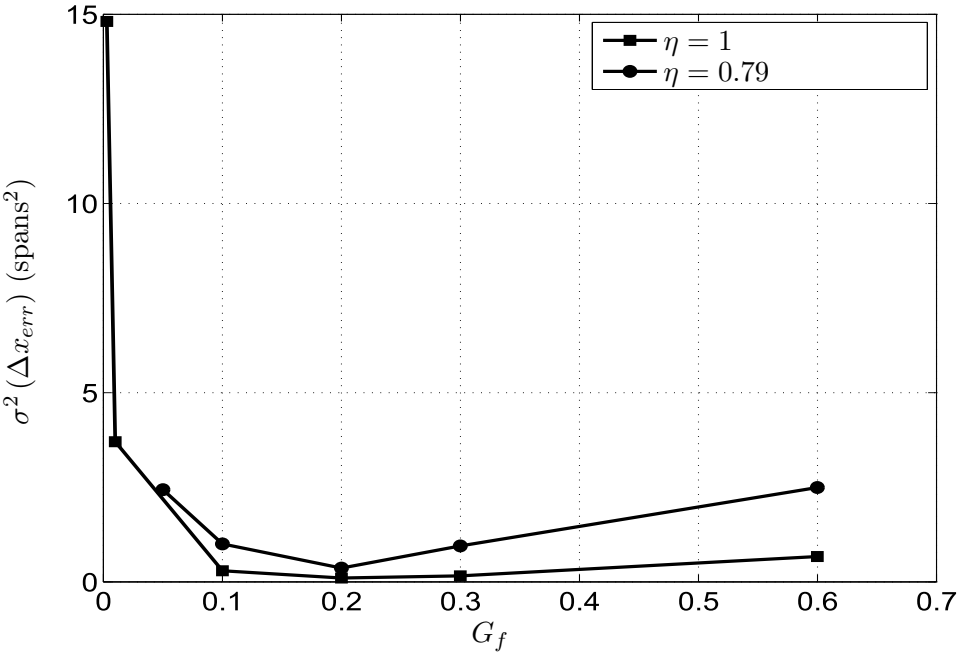


Figure 6.10: Results of separation error variance at different lateral separations and control gains, when submersed in moderate turbulence

atmosphere at larger lateral separations. This is because the trail aircraft is better able to maintain its station, due to the less aggressive fluctuations in benefit experienced further outboard of the leader. Nonetheless, the trailing aircraft still experiences a greater reduction in fuel consumption when flying at the "sweet spot", but at a cost of station-keeping precision. It is also noted, from Fig. 6.7, that at very low gains, the fuel consumption starts to increase dramatically. This is because the engines response is too slow to maintain the trail aircraft's station within the formation, but just fast enough to have kept it within the station-keeping tolerance (set at a liberal 20 spans) for the duration of the one-hour flight test. In other words, the trail aircraft is unable to keep up with the leader, and slowly falling behind, is constantly increasing its throttle to a point of higher fuel-burn.

At both lateral separations the optimum fuel saving, with a corresponding optimum precision, appears to occur at the test point $G_f = 0.2$. Here, the fuel saving is approximately 34.5% and 24.9% for $\eta = 0.79$ and $\eta = 1$, respectively. The corresponding station-keeping precisions, as represented by the standard deviations in Fig. 6.9, are approximately ± 0.6 and ± 0.3 spans, respectively. At $\eta = 1$, the mean separation is found to closely follow the desired nominal stream-wise separation of 20 spans. However, at $\eta = 0.79$ this tracking becomes less accurate, with mean values found to be approximately 1.2 and 0.04 spans off of the desired nominal stream-wise separation for $\eta = 0.79$ and $\eta = 1$ respectively. This mean error is because of the steady-state tracking error inherent of a proportional gain controller, and is larger at $\eta = 0.79$ because of the amplified formation dynamics. Finally, the lateral separation at $\eta = 0.79$ appears to have a narrower optimised gain bandwidth, than that of the separation at $\eta = 1$. This indicates that the "sweet spot" is a more restrictive area to fly in during moderate turbulence, but becomes less restrictive in light turbulence.

Chapter 7

Conclusions

Unfortunately, due to the computational expense of the formation simulations, the resulting data from the simulations are sparsely scattered and further tests are required to identify more distinct trends. However, even when including the effects of dynamic throttling, it has been clearly shown that significant fuel saving benefits may be achieved by a twin-aircraft formation in moderate turbulence. Interestingly, it was found that the optimum station-keeping precision corresponds to optimal savings in fuel. This suggests that better streamwise station-keeping precision improves not only the practicality and safety of the formation, but also the associated fuel savings. Flying at the "sweet spot" results in much higher fuel savings (increased by approximately 10%), but at a cost of station-keeping precision. Therefore, in this zone, the trailing aircraft's movements are more erratic, and poses questions about the practicality of flying in such regions. Further outboard, the trailing aircraft's movements are more stable and, despite a significant decrease in fuel savings, offer improved practicality in-terms of safety and formation control. It is therefore concluded that a fuel saving of approximately 25% may be achieved at a practical lateral separation of 1 span, corresponding to a stream-wise separation of 20 $\pm$0.3 spans.

Chapter 8

Recommendations

The achievable fuel savings appear to be highly dependent on the manner by which the trail aircraft is controlled. Therefore subsequent investigations could, initially, also assume perfect control of the aircraft's lateral motion, but with the inclusion of control surface deflections required for lateral trim. This may require the use of a higher fidelity aerodynamics model. This will allow for a better approximation of the aerodynamic state of each aircraft within the formation. The altitude constraint on the aircraft's decoupled longitudinal motion should be relaxed, and a control system implemented to manage the both the altitude and streamwise position of each aircraft within the formation, which may be achieved through a combination of throttle and elevator actuations. Subsequently, it is likely that the engine's contribution to maintaining altitude will directly impact the achievable fuel consumption. It is suggested that this 3DOF control system be based on existing autopilot systems, with the adaptation of stream-wise separation control for the trailing aircraft. Furthermore, it is recommended that a reduced order engine model be used to alleviate the mathematical stiffness of the entire formation model, and improve simulation speeds. Post simulation analysis using the high-fidelity engine model may be used, thereafter, to infer any detrimental effects of dynamic throttling. Finally, further studies could investigate the implications of string stability on a formation with more than two aircraft as well as fuel savings at various nominal stream-wise separations.

Appendix A

Induced Flow Calculations

This appendix details the calculations of the flows induced at a bound vortex of the trailing aircraft by the trailing vortices of the lead aircraft. It is noted that these derivations follow those presented by Bizinos [27], and have been included in this appendix for the sake of completeness and ease of reference.

First, the derivation of the flows induced due to the Biot-Savart law, applying the Helmholtz velocity profile, are considered. This is done for demonstrative purposes (the result is not implemented in this work), as it is simpler to visualise geometrically. The results of this derivation are used to aid the derivation of the induced flows resulting from the Burnham Hallock profile. First, the flows induced by the vortex nearest the trail aircraft are considered, and then the induced flow of the furthest trailing vortex. These results are then combined to obtain the overall induced flow.

Figure A.1 is a schematic of a twin-aircraft formation, with the trailing aircraft flying behind and below the lead. This schematic includes a top and rear view of the formation, to better visualise the geometric separation between the aircraft. Each aircraft is represented by a horseshoe vortex, depicted in light grey, and for this explanation, the tailplane horseshoe vortex of the trailing aircraft has been omitted. The Biot-Savart law was explained in Section 3.1.3 with Reference to Figure 3.3. Figure 3.3, has effectively been superimposed onto Figure A.1, such that the geometry relating to the induced flow may be discussed.

In Section 3.1.3, the magnitude of the induced flow, using the Helmholtz profile, was derived and presented in Equation 3.7, which is repeated here for convenience:

$$V_{\theta_H} = \frac{\Gamma}{4\pi h}\left(1 + \cos\theta\right) \tag{A.1}$$

To solve this equation, it is necessary to define the geometric terms h and $\cos\theta$. Using the center of the trailing aircraft's bound vortex as a Cartesian reference point, the variable y is defined to locate point P along this bound

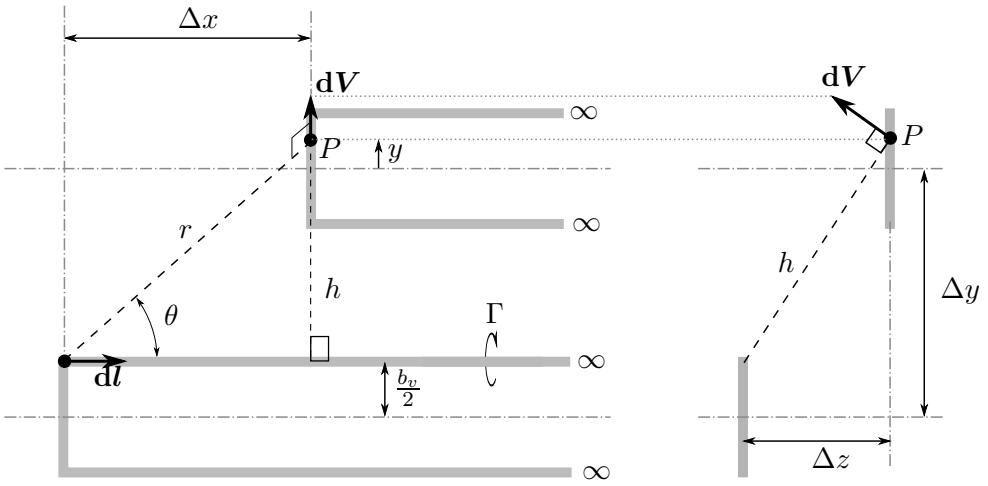


Figure A.1: Flow induced by the "near" trailing vortex, using the Helmholtz vortex profile

vortex. From Figure A.1, the perpendicular distance between the trailing vortex and point P is:

$$h = \sqrt{\left(y + \Delta y - \frac{b_v}{2}\right)^2 + \Delta z^2} \tag{A.2}$$

And the angular term $\cos\theta$ may be expressed as:

$$\cos\theta = \frac{\Delta x}{\sqrt{h^2 + \Delta x^2}} \tag{A.3}$$

Substituting Equations A.2 and A.3 back into Equation A.1, and rearranging, yields:

$$\frac{4\pi}{\Gamma_j} V_{\theta_{near,kH}} = \left(\frac{1}{\sqrt{\left(y + \Delta y - \frac{b_v}{2}\right)^2 + \Delta z^2}}\right)\left(1 + \frac{\Delta x}{\sqrt{\left(y + \Delta y - \frac{b_v}{2}\right)^2 + \Delta z^2 + \Delta x^2}}\right) \tag{A.4}$$

Equation A.4, describes the flow induced by the lead aircraft's trailing vortex positioned closest to the trailing aircraft. The notation of this equation follows that defined by Bizinos [27], where j and k represent the lead and trail aircraft respectively.

However, we are mainly concerned with the component of flow that effects the downwash of the trailing aircraft, i.e. the vertical component in the inertial reference frame assuming straight and level flight. Figure A.2, de-

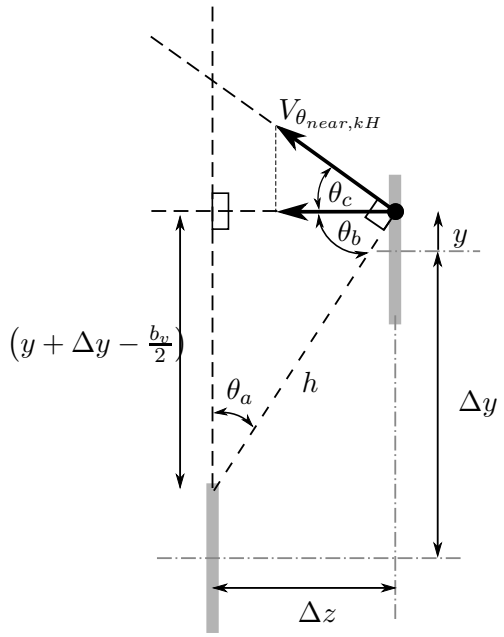


Figure A.2: Downwash component of the induced flow

picts the "Rear View" of Figure A.1 along with some additional construction lines and variables. From this figure, the downwash is given as:

$$w_{near,kH} = V_{\theta_{near,kH}} \cos\theta_c \tag{A.5}$$

It may also be seen that $\theta_a = \theta_c$, because $\theta_a + \theta_b = \pi/2$ and $\theta_c + \theta_b = \pi/2$, therefore:

$$\cos\theta_c = \cos\theta_a = \frac{y + \Delta y - \frac{b_v}{2}}{h} \tag{A.6}$$

Next, Equations A.4 and A.6 may be substituted into Equation A.5 to yield the magnitude of the downwash induced on the bound vortex, which is given by Equation A.7. It is noted that downwash has been defined as positive in the vertically downward direction, and hence the negative sign in Equation A.7.

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{near,kH} = -\left(\frac{1}{\sqrt{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2}}\right)\ldots \left(1+\frac{\Delta x}{\sqrt{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2+\Delta x^2}}\right)\ldots \left(\frac{y+\Delta y-\frac{b_v}{2}}{\sqrt{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2}}\right) \tag{A.7}$$

The downwash induced by the "far" trailing vortex may be calculated in a similar fashion, and therefore its derivation has not been included here. The difference between the two scenarios is that the circulation of the far vortex acts in the opposite direction to that of the near vortex. In addition, the lateral spacing between the far trailing vortex and the bound vortex of the trailing aircraft, is increased by a span b_v. Equation A.8 gives the downwash induced by the far trailing vortex, and these differences are evident from the absence of the preceding negative sign, as well as the replacement of the $-\frac{b_v}{2}$ term with a positive $+\frac{b_v}{2}$ term.

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{far,kH} = \left(\frac{1}{\sqrt{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2}}\right)\ldots \left(1+\frac{\Delta x}{\sqrt{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2+\Delta x^2}}\right)\ldots \left(\frac{y+\Delta y+\frac{b_v}{2}}{\sqrt{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2}}\right) \tag{A.8}$$

Next, the total approximate downwash induced at the trailing aircraft, may be obtained via the addition of Equations A.7 and A.8:

$$w_{jk,H} = w_{near,kH} + w_{far,kH} \tag{A.9}$$

where the subscript jk denotes the influence of the lead aircraft j on the trail aircraft k. However, the above summation, in its expanded form, is rather convoluted. Recalling Munk's Stagger Theorem [16] and that the trailing vortices extend to infinity downstream, the flow induced will be equivalent no matter the streamwise separation between aircraft. Therefore, the expanded summation of Equations A.7 and A.8, may be drastically simplified by allowing $\Delta x \to \infty$. This allowance results in the second term

in the middle set of brackets (on the right hand side of Equations A.7 and A.8) to equal unity. Therefore, the summation simplifies to:

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{jk,H} = -\left(\frac{2}{\sqrt{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2}}\right)\left(\frac{y+\Delta y-\frac{b_v}{2}}{\sqrt{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2}}\right)... + \left(\frac{2}{\sqrt{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2}}\right)\left(\frac{y+\Delta y+\frac{b_v}{2}}{\sqrt{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2}}\right) \tag{A.10}$$

Simplifying further:

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{jk,H} = 2\left[\frac{\left(y+\Delta y+\frac{b_v}{2}\right)}{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2} - \frac{\left(y+\Delta y-\frac{b_v}{2}\right)}{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2}\right] \tag{A.11}$$

Thus, Equation A.11 is used to calculate the downwash induced at a point on the trailing aircraft's bound vortex, in the presence of two trailing vortices from a lead aircraft. An important result of this equation, is that the right hand side is completely defined by the geometry of the formation.

Now, Equation A.11 can be easily modified to include the viscous core specified by the Burnham Hallock profile, with the result shown by Equation A.12. Here, the downwash has the subscript BH to indicate that it includes the viscous core of the Burnham Hallock profile.

$$\left(\frac{4\pi}{\Gamma_j}\right) w_{jk,BH} = 2\left[\frac{\left(y+\Delta y+\frac{b_v}{2}\right)}{\left(y+\Delta y+\frac{b_v}{2}\right)^2+\Delta z^2+r_c^2} - \frac{\left(y+\Delta y-\frac{b_v}{2}\right)}{\left(y+\Delta y-\frac{b_v}{2}\right)^2+\Delta z^2+r_c^2}\right] \tag{A.12}$$

Appendix B

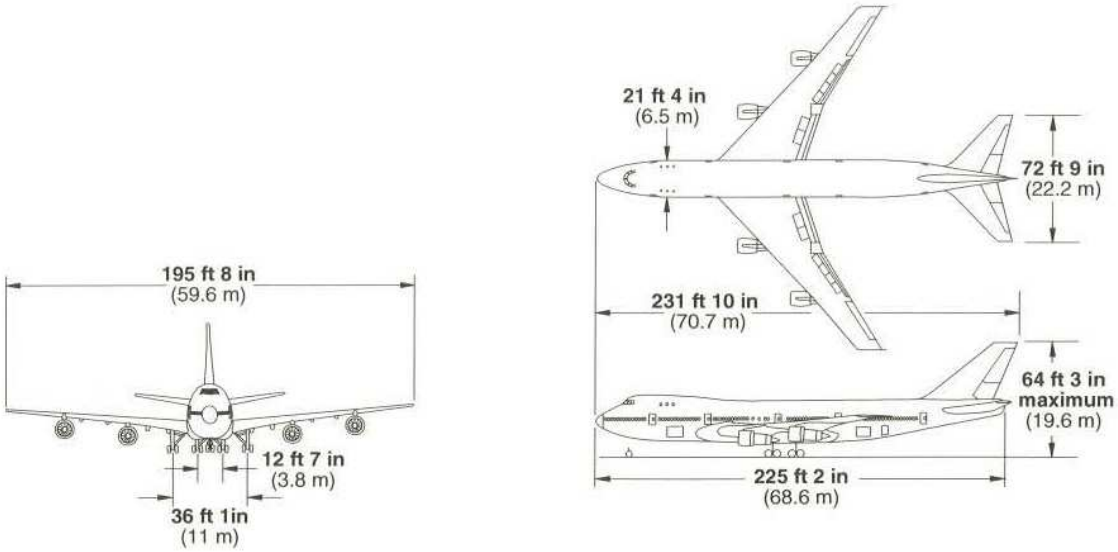
Aircraft Properties

All aircraft models presented in this work, have been based on the Boeing 747-100. This aircraft was selected due to the abundance of data available in the public domain. In addition, many aircraft of the Boeing 747 family are still in service, and in fact production, today. Therefore, the aircraft models developed in this work, which based on 747-100 data, are deemed to be representative of a typical commercial airliner.

A diagram of the Boeing 747-100, taken from the Boeing website [114], is depicted in Figure B.1. This diagram displays information about the aircraft's geometry and weight. Additional information regarding inertias, aerodynamic derivatives, control characteristics and stability augmentation systems, has been obtained from Heffley and Jewell [115]. It is noted that this work has only considered the longitudinal motion of the aircraft, and therefore only information relating to the longitudinal motion is presented here.

Table B.1, presents the dimensional, mass and flight condition parameters extracted from Heffley and Jewell for a selected cruise flight condition. Only flight at cruise is considered in this work, and therefore no data on other flight conditions has been included in this document. In addition, it is noted that the values presented in this table supersede any values defined in Figure B.1, as they pertain to a particular flight condition and are therefore, in that context, more accurate.

Table B.2, gives the longitudinal motion aerodynamic coefficients, corresponding to the flight conditions of Table B.1. The majority of these coefficients were extracted from Heffley and Jewell, and, where not available, calculated according to trim conditions.



Max takeoff weight	lb	710,000 or 712,000 or 733,000 or 735,000				
Max landing weight	lb	564,000 or 585,000*				
Max zero fuel weight	lb	526,500 or 545,000*				
Fuel capacity	U.S. gal	47,210 or 48,445				
Lower hold cargo	ft^3	30 LD-1 containers	5,190	or	5 pallets + 14 LD-1s	4,497
		Bulk	1,000		Bulk	1,000
		Total	6,190		Total	5,497

* Conversion to all-cargo model

Figure B.1: Diagram of the Boeing 747-100 Characteristics. Reprinted from the Boeing Website [114]

Characteristic	Parameter	Units	Value
Flight Conditions	H	(m)	12192
	α	(deg)	4.6
	M	$(-)$	0.8
Mass and Inertia	m	(kg)	288773.232
	I_y	$\left(kg.m^2\right)$	44.878×10^6
Main Geometry	S	$\left(m^2\right)$	510.967
	b	(m)	59.643
	$\bar{c}$	(m)	8.324
	c.g.	(m)	$0.25\bar{c}$
	a.c.	(m)	$0.25\bar{c}$
Tailplane Geometry	S_{tp}	$\left(m^2\right)$	141
	b_{tp}	(m)	22.4
	l_{tp}	(m)	29.6
	z_{tp}	(m)	-5.1
Engines	i_{th}	(deg)	2.5
	l_{th}	(m)	3.048

Table B.1: Boeing 747-100 dimensional, mass and flight condition parameters. Extracted from Heffley and Jewell [115]

Coefficient	Trim	$\frac{\partial}{\partial \alpha}$	$\frac{\partial}{\partial \dot{\alpha}}$	$\frac{\partial}{\partial M}$	$\frac{\partial}{\partial \delta_e}$	$\frac{\partial}{\partial q}$
C_D	0.04414	0.43287	0.00000	0.02647	0.00000	0.00000
C_L	0.64986	4.91632	5.91000	0.14694	0.36629	6.00000
C_m	0.00000	-1.03670	-0.11223	0.17184	-1.44022	-0.42266

Table B.2: Boeing 747-100 longitudinal aerodynamic coefficients (rounded to 5 decimal places). Extracted from Heffley and Jewell [115]

Appendix C

Gas Properties

C.1 Theory

The working medium in an engine is gaseous, and it is important that the thermodynamic properties of the gases be calculated accurately for different gas compositions and conditions.

The thermodynamic properties of a gas and its comprising species, tend to vary with changes in temperature and pressure. Such variations are also unique to each species, according to its atomic make-up. Such relations are typically represented in look-up tables, as in the case of the JANAF tables (e.g. Reference [116]), or alternatively as polynomial curve-fits.

Polynomial curve-fits are generally preferred in computations, because of the speed penalties encountered with interpolations of look-up tables. The NASA Glenn Research Center has been developing a library of polynomial curve-fits to facilitate the calculation of thermodynamic properties of different species over various temperature ranges. This ongoing work has been comprehensively documented by McBride, Zehe and Gordon [117], and is the source upon which this section is based.

For an individual specie, the specific heat, enthalpy and entropy may be calculated, as a function of temperature, from the empirical formulae given in Equations C.1, C.2 and C.3, respectively.

$$\frac{c_{p,mol}(T)}{R} = a_1 T^{-2} + a_2 T^{-1} + a_3 + a_4 T + a_5 T^2 + a_6 T^3 + a_7 T^4 \qquad \text{(C.1)}$$

$$\frac{h_{mol}(T)}{RT} = -a_1 T^{-2} + a_2 \frac{\ln(T)}{T} + a_3 + a_4 \frac{T}{2} + a_5 \frac{T^2}{3} + a_6 \frac{T^3}{4} + a_7 \frac{T^4}{5} + \frac{b_1}{T} \qquad \text{(C.2)}$$

$$\frac{s_{mol}(T)}{R} = -a_1 T^{-2} - a_2 T^{-1} + a_3 \ln(T) + a_4 T + a_5 \frac{T^2}{2} + a_6 \frac{T^3}{3} + a_7 \frac{T^4}{4} + b_2 \qquad \text{(C.3)}$$

The coefficients, denoted by a and b, are constants which differ across species and temperature ranges. McBride *et al.* [117], have documented a library containing the values of the coefficients for numerous species over large temperature ranges. The coefficients in this library, yield specific heat, enthalpy and entropy in units of ${}^{kJ}/_{kmol}K$, ${}^{kJ}/_{kmol}$ and ${}^{kJ}/_{kmol}K$ respectively. For a single specie, these parameters may be converted to mass specific units, through dividing by the molecular weight of that particular specie:

$$c_p(T) = \frac{c_{p,mol}(T)}{\text{MW}},\ {}^{kJ}/_{kg}K \tag{C.4}$$

$$h(T) = \frac{h_{mol}(T)}{\text{MW}},\ {}^{kJ}/_{kg} \tag{C.5}$$

$$s(T) = \frac{s_{mol}(T)}{\text{MW}},\ {}^{kJ}/_{kg}K \tag{C.6}$$

The specific heats ratio may then be calculated in one of two ways; from the mol specific heat and universal gas constant, or from the mass specific heat and that particular specie's gas constant (this is simply the universal gas constant divided by the molecular weight of the specie, R_u/MW, which is obtained from the afore mentioned library). Either combination may be used and a general form of its calculation is given as:

$$\gamma = \frac{c_p}{c_p - R} \tag{C.7}$$

However, as is often the case in gas turbine engines, the gas is not comprised of a single specie but of numerous species. In addition, the amount of each specie within the mixture will differ according to the engine's operation. It is therefore, important to weight each specie's contribution to the thermodynamic properties of the mixture according to the relative amount of that specie present in the mixture. The mole and mass fractions are commonly used for this purpose, however it is the former which is utilised in this particular work. The mole fraction is simply a ratio of the number of moles of a particular specie i to that of the entire mixture, $mf_i = {}^{\text{mol}_i}/_{\text{mol}_{mix}}$. The mole fraction may then be used to calculate the contribution of each specie, and therefore the thermodynamic properties of a gas mixture of n species:

$$X_{mix} = \sum_{i}^{n} (mf_i \times X_i) \tag{C.8}$$

where X may be substituted for the parameter of interest. It must be noted that the above equation must be used to determine the gas mixture's gas constant (used in calculating the specific heats ratio) and molecular weight (used in converting to mass specific values from mole specific values).

C.2 MATLAB Routines

For the engine model developed in this work, two libraries were used in determining the gas properties. The first, to calculate the thermodynamic properties of air (upstream of combustion) and the second to calculate the properties of a mixture of combustion products (flowing downstream of combustion). Two routines were developed, instead of one, because air is obtained from the NASA library as a single specie. Therefore, the calculations of gas properties upstream of the combustion are significantly simpler than that of the downstream calculation. The separate routines, thus, allow for faster computations by avoiding unnecessary searching through libraries containing species of possible combustion products, as well as mixture calculations. The two routines are presented below.

C.2.1 Air Properties

The routine developed for determining the thermodynamic properties of air, is presented in Code Listing C.1. This routine takes in the temperature of the air and returns mass specific heat, specific heats ratio, enthalpy, entropy and gas constant. Polynomial curve-fit coefficients are supplied over two temperature ranges, ($200K \leq T \leq 1000K$) and ($1000K \leq T \leq 6000K$), which are more than sufficient to cover the operating temperatures of the engine. However, should the air temperature exceed these temperatures, an error is encountered and a message detailing the error is displayed to the command window.

Code Listing C.1: MATLAB Function AirTherm.m for determining the thermodynamic properties of air at different temperatures

```
1  %=============================================================
2  %            Calculate Thermodynamic Properties of Air
3  %=============================================================
4  %
5  % NAME:         AirTherm.m
6  %
7  % AUTHOR:       D. Sanders, University of Cape Town
8  %
9  % DATE:         15 August 2012
10 %
11 % FUNCTION:     function [CP, Y, H, S, Rs] = AirTherm (T)
12 %
13 % PURPOSE:      Calculates the thermodynamic properties of ...
       air at various temperatures using polynomial curvefits.
14 %
15 % INPUTS:
16 %
17 %   T           = Air Temperature                          [K]
18 %
```

```
19 % OUTPUTS:
20 %
21 %   CP          = specific heats                    [kJ/kg-K]
22 %   Y           = specific heats ratio                  [-]
23 %   H           = Enthalpy                           [kJ/kg]
24 %   S           = Entropy                           [kJ/kg-K]
25 %   Rs          = Mixture specific gas constans     [kJ/kg-k]
26 %
27 % SUBROUTINES:  None
28 %
29 % REFERENCES:
30 %
31 %   Data:       McBride, Zehe & Gordon, "Coefficients for ...
       Calculating Thermodynamic Properties of Individual ...
       Species" National Aeronautics and Space Administration, ...
       John H. Glenn Research Center at Lewis Field, 2002
32 %
33 %   Formulae:   Turns, S. "An Introduction to Combustion: ...
       Concepts and Applications" McGraw-hill New York, 1996
34 %
35 % For coding conventions, please refer to:
36 %
37 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
38 %   ver 1.5. October 2002.
39 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
40 %===============================================================
41
42 function [CP, Y, H, S, Rs] = AirTherm (T)
43
44 %===============================================================
45
46 %% Properties of Air
47
48 % Molecular Weight          [kg/kmol]
49 MW = 28.96512;
50
51 % Universal Gas Constant    [kJ/kmol-K]
52 R = 8.314510;
53
54 % Specific Gas Constant     [kJ/kg-K]
55 Rs = 0.2871;
56
57 %% Select Polyfit Coefficients
58
59 if (T ≥ 200) && (T ≤ 1000)
60 %===============================================================
61 a = [1.009950160D+04   -1.968275610D+02    5.009155110D+00 ...
62     -5.761013730D-03    1.066859930D-05   -7.940297970D-09 ...
63      2.185231910D-12   -1.767967310D+02   -3.921504225D+00];
64
65 elseif (T > 1000) && (T ≤ 6000)
66
67 a = [2.415214430D+05   -1.257874600D+03    5.144558670D+00 ...
68     -2.138541790D-04    7.065227840D-08   -1.071483490D-11 ...
```

```
69      6.577800150D-16     6.462263190D+03    -8.147411905D+00];
70
71  else
72      error('Temperature of air falls outside of curvefit range')
73  end
74
75  %% Calculate Thermodynamic Properties
76
77  % Specific Heat         [kJ/kmol-K]
78  cp = R * ( a(1)*(T^(-2)) + a(2)*(T^(-1)) + a(3) + a(4)*(T) ...
        + a(5)*(T^2) + a(6)*(T^3) + a(7)*(T^4) );
79
80  % Enthalpy              [kJ/kmol]
81  h = R * T * ( -a(1)*(T^(-2)) + a(2)*log(T)/T + a(3) + ...
        a(4)*T/2 + a(5)*(T^2)/3 + a(6)*(T^3)/4 + a(7)*(T^4)/5 + ...
        a(8)/T );
82
83  % Entropy               [kJ/kmol-K]
84  s = R * ( -a(1)*(T^(-2))/2 - a(2)*(T^(-1)) + a(3)*log(T) + ...
        a(4)*(T) + a(5)*(T^2)/2 + a(6)*(T^3)/3 + a(7)*(T^4)/4 + ...
        a(9) );
85
86  % Convert to Mass Specific Properties Final mixture properties
87  % Specific heat         [kJ/kg-K]
88  CP = cp / MW;
89
90  % Specific heat ratio   [-]
91  Y = cp/(cp - R);
92
93  % Specific enthalpy     [kJ/kg]
94  H = h / MW;
95
96  % Specific enthalpy     [kJ/kg]
97  S = s / MW;
98
99  % Convert to Standard Units
100     % CP: [kJ/kg-K] --> [J/kg-K]
101     CP = 1000 * CP;
102
103     % H: [kJ/kg] --> [J/kg]
104     H = 1000 * H;
105
106     % S: [kJ/kg] --> [J/kg]
107     S = 1000 * S;
108
109     % Rs: [kJ/kg-K] --> [J/kg-K]
110     Rs = 1000 * Rs;
111
112 return
```

C.2.2 Mixture Properties

The routine developed for determining the thermodynamic properties of a mixture of combustion products, is presented in Code Listing C.2 [1]. The routine takes in values of fuel temperature, air/mixture temperature and a vector of mole fractions, and returns the mass specific heat, specific heats ratio, enthalpy, entropy and gas constant of the mixture. The possible species included in the routine are: Fuel (JET-A), Oxygen O_2, Nitrogen N_2, Carbon Dioxide CO_2 and Water H_2O. The library has been limited to these particular species, as the combustion model is based on adiabatic flame temperature calculations which ignore the effects of dissociation. In addition, the fuel is assumed to be injected in a liquid state and under a temperature of 550K. Thereafter, any fuel present in the mixture, post combustion, is assumed to be gaseous. Code Listing C.2, includes coefficients which are valid over different temperature ranges (which are documented in the code's comments). As before, if these temperature ranges are exceeded the user will receive an error message in the command window detailing the error. The mole fraction imputed, is used to calculate overall mixture properties as described in the previous section.

Code Listing C.2: MATLAB Function CombTherm.m for determining the thermodynamic properties of gas mixtures at different temperatures and compositions

```
1  %=============================================================
2  %     Calculate Thermodynamic Properties of Gas Mixtures
3  %=============================================================
4  %
5  % NAME:         CombTherm.m
6  %
7  % AUTHOR:       D. Sanders, University of Cape Town
8  %
9  % DATE:         15 October 2012
10 %
11 % FUNCTION:     [CP, Y, H, S, Rs] = CombTherm (Tfuel, Tmix, N)
12 %
13 % PURPOSE:      Calculates the thermodynamic properties of ...
       a gas mixture atvarious temperatures using polynomial ...
       curvefits. The mixture contains quantities of ...
       Fuel(Jet-A), O2, N2, CO2 and H2O.
14 %
15 % INPUTS:
16 %
17 %   Tfuel       = Fuel Temperature                          [K]
18 %   Tmix        = Temperature of air and/or mixture         [K]
19 %   N           = Mole quantity vector:
```

[1] The actual library data has been omitted from the code listing to save space within this document

```
20 %                 [Fuel; O2; N2; CO2; H2O]                [mol]
21 %
22 % OUTPUTS:
23 %
24 %   CP          = Specific heats                       [kJ/kg-K]
25 %   Y           = Specific heats ratio                      [-]
26 %   H           = Enthalpy                               [kJ/kg]
27 %   S           = Entropy                                   [-]
28 %   Rmix        = Mixture specific gas constans        [kJ/kg-k]
29 %
30 % SUBROUTINES:  None
31 %
32 % REFERENCES:
33 %
34 %   Data:       McBride, Zehe & Gordon, "Coefficients for ...
       Calculating Thermodynamic Properties of Individual ...
       Species" National Aeronautics and Space Administration, ...
       John H. Glenn Research Center at Lewis Field, 2002
35 %
36 %   Formulae:   Turns, S. "An Introduction to Combustion: ...
       Concepts and Applications" McGraw-hill New York, 1996
37 %
38 % For coding conventions, please refer to:
39 %
40 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
41 %   ver 1.5. October 2002.
42 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
43 %===============================================================
44
45 % ROW ORDER OF SPECIES MATRICES AND VECTORS:
46     % 1.    FUEL
47     % 2.    O2
48     % 3.    N2
49     % 4.    CO2
50     % 5.    H2O
51
52 %===============================================================
53
54 function [CP, Y, H, S, Rs] = CombTherm (Tfuel, Tmix, N)
55
56 %===============================================================
57 %% Thermodynamic Data of species
58
59 % Molecular weights ...
                                                        [kg/kmol]
60 MW = [167.3110200; 31.999; 28.013; 44.011; 18.016];
61
62 %% Jet-A: C12H23
63
64 % The polynomial fits may only be used to predict ...
       enthalpies of reactants. It is assumed that fuel will ...
       enter the combustion chamber as a liquid and under a ...
       temperature of 550K
65
```

```
% Liquid: Temp range 220.000 - 550.000 K
if (Tfuel ≥ 220) && (Tfuel ≤ 550)
a(1,:) = [...];

% Gas: Temp range 273.150 - 1000.000 K
elseif (Tfuel ≥ 550) && (Tfuel ≤ 1000)

a(1,:) = [...];

% Gas: Temp range 1000.000 - 6000.000 K
elseif (Tfuel > 1000) && (Tfuel ≤ 6000)

a(1,:) = [...];

% Operation outside of temperature range
else
    error('Fuel temperature outside of polyfit range')
    return
end

%% Non-Fuel Species
%=============================================================
% Temp range 200.000 - 1000.000 K
if (Tmix ≥ 200) && (Tmix ≤ 1000)
a(2:5,:) = [...];

% Temp range 1000.000 - 6000.000 K
elseif (Tmix > 1000) && (Tmix ≤ 6000)
a(2:5,:) = [...];

% Temp range 6000.000 - 20000.000 K
% H2O not valid here
elseif (Tmix > 6000) && (Tmix ≤ 20000)
    error('H2O temperature outside of polyfit range')
a(2:5,:) = [...];
    return

% Operation outside of temperature range
else
    error('Mixture temperature outside of polyfit range')
    return
end

%% Enthalpy Calculations
% Universal Gas Constant    [kJ/kmol-K]
R = 8.314510;

% Fuel
% Specific Heat             [kJ/kmol-K]
cp(1,1) =   R * ( a(1,1)*(Tfuel^(-2)) + a(1,2)*(Tfuel^(-1)) ...
    + a(1,3) + a(1,4)*(Tfuel) + a(1,5)*(Tfuel^2) + ...
    a(1,6)*(Tfuel^3) + a(1,7)*(Tfuel^4) );

% Enthalpy                  [kJ/kmol]
```

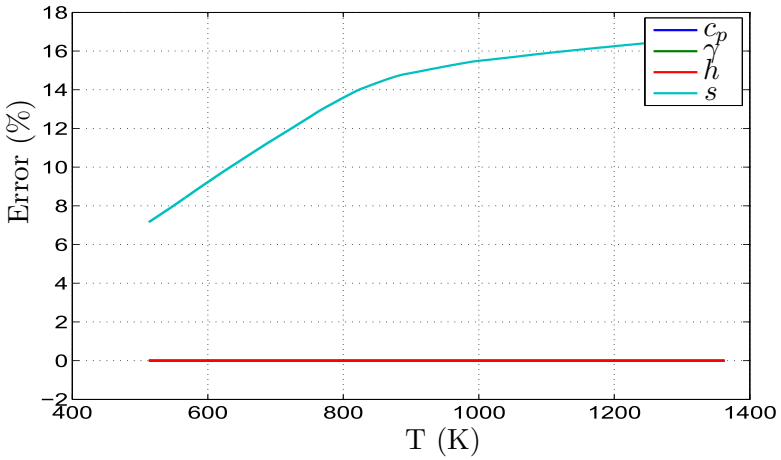
```
118 h(1,1) =    R * Tfuel * ( -a(1,1)*(Tfuel^(-2)) + ...
        a(1,2)*log(Tfuel)/Tfuel + a(1,3) + a(1,4)*(Tfuel)/2 + ...
        a(1,5)*(Tfuel^2)/3 + a(1,6)*(Tfuel^3)/4 + ...
        a(1,7)*(Tfuel^4)/5 + a(1,8)/Tfuel );
119
120 % Entropy                   [kJ/kmol-K]
121 s(1,1) = R * ( -a(1,1)*(Tfuel^(-2))/2 - a(1,2)*(Tfuel^(-1)) ...
        + a(1,3)*log(Tfuel) + a(1,4)*(Tfuel) + ...
        a(1,5)*(Tfuel^2)/2 + a(1,6)*(Tfuel^3)/3 + ...
        a(1,7)*(Tfuel^4)/4 + a(1,9));
122
123 % Other
124 % Specific Heat             [kJ/kmol-K]
125 cp(2:5,1) = R * ( a(2:5,1)*(Tmix^(-2)) + ...
        a(2:5,2)*(Tmix^(-1)) + a(2:5,3) + a(2:5,4)*(Tmix) + ...
        a(2:5,5)*(Tmix^2) + a(2:5,6)*(Tmix^3) + ...
        a(2:5,7)*(Tmix^4) );
126
127 % Enthalpy                  [kJ/kmol]
128 h(2:5,1) =  R * Tmix * ( -a(2:5,1)*(Tmix^(-2)) + ...
        a(2:5,2)*log(Tmix)/Tmix + a(2:5,3) + a(2:5,4)*(Tmix)/2 ...
        + a(2:5,5)*(Tmix^2)/3 + a(2:5,6)*(Tmix^3)/4 + ...
        a(2:5,7)*(Tmix^4)/5 + a(2:5,8)/Tmix );
129
130 % Entropy                   [kJ/kmol-K]
131 s(2:5,1) = R * ( -a(2:5,1)*(Tmix^(-2))/2 - ...
        a(2:5,2)*(Tmix^(-1)) + a(2:5,3)*log(Tmix) + ...
        a(2:5,4)*(Tmix) + a(2:5,5)*(Tmix^2)/2 + ...
        a(2:5,6)*(Tmix^3)/3 + a(2:5,7)*(Tmix^4)/4 + a(2:5,9) );
132
133 %% Final mixture properties
134 % Define mole fractions     [mol], [-]
135 Nmix        = sum(N);
136 Nf          = N/Nmix;
137
138 % Molecular Weight          [kg/kmol]
139 MWmix       = sum(Nf.*MW);
140
141 % Specific heat             [kJ/kmol-K], [kJ/kg-K]
142 cpmix_mol   = sum(Nf.*cp);
143 CP          = cpmix_mol / MWmix;
144
145 % Specific heat ratio       [kJ/kg-K], [-]
146 Rmix        = R/MWmix;
147 Y           = CP/(CP - Rmix);
148
149 % Specific enthalpy         [kJ/kmol], [kJ/kg]
150 hmix_mol    = sum(Nf.*h);
151 H           = hmix_mol / MWmix;
152
153 % Total Entropy             [kJ/kmol-K], [kJ/kg-K], [kJ/K]
154 smix_mol   = sum(Nf.*s);
155 S           = smix_mol / MWmix;
156
```

```
157 % Convert to Standard Units
158     % CP: [kJ/kg-K] --> [J/kg-K]
159     CP = 1000 * CP;
160
161     % H: [kJ/kg] --> [J/kg]
162     H = 1000 * H;
163
164     % S: [kJ/kg] --> [J/kg]
165     S = 1000 * S;
166
167     % Rs: [kJ/kg-K] --> [J/kg-K]
168     Rs = 1000 * Rmix;
169 return
```

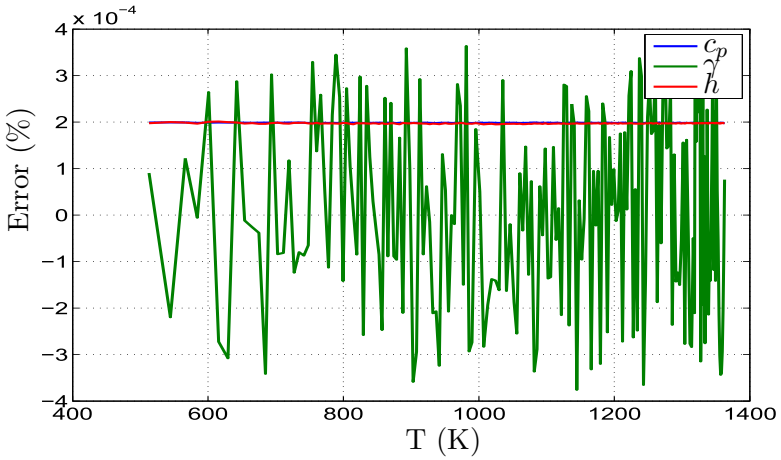
C.3 Validation

The polynomial curve-fits presented above, have been validated and verified at the NASA Glenn Research Center [117]. However, it is necessary to compare the outputs between GSP and the routines described above. This serves the purpose of identifying any discrepancies which may raise questions about the validity of the component models developed in this work. A comparison between GSP and the routine AirTherm is depicted in Figure C.1. As is evident from Figure C.1a, there is a very large discrepancy in entropy values. The reason for this is unknown to the author, and fortunately this is not of major concern as entropy is not used in any of the components calculations. Removing entropy from the comparison yields Figure C.1b. These parameters, c_p, h and γ, are used in component calculations and are of high importance. As is evident, there are very small errors, thus indicating a good correlation in the prediction of these parameters. Figure C.1c removes γ from the comparison to reveal how closely the two models agree with one another. As already mentioned, the only concern is the discrepancy in the entropy predicted. This error has been ignored because entropy does not effect any of the engine models calculations and therefore does not effect the validity of this work.

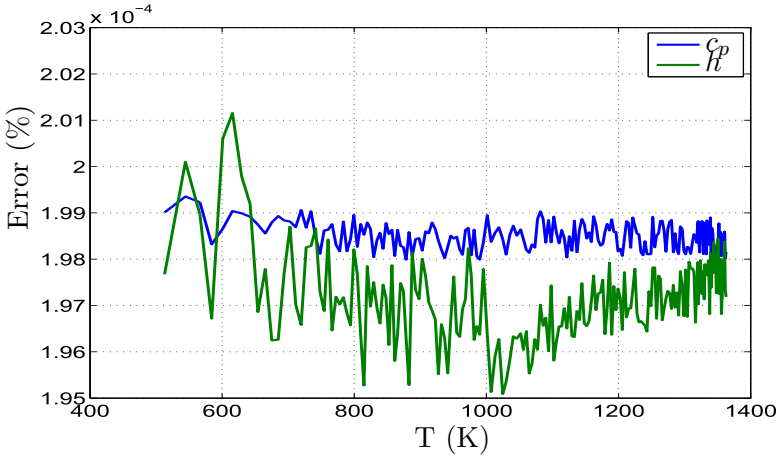
Unfortunately, it is very difficult to compare the results of predictions in mixtures between the model in GSP and that developed in this appendix. This is because the gas composition obtained from GSP uses a more comprehensive composition of air (i.e. Argon), accounting for additional species. In addition, GSP also includes species which result from dissociation effects. Dissociation and the less significant species of air have not been incorporated in the engine model developed in this work, and therefore the 'CombTherm.m' routine developed cannot accommodate the compositions specified by GSP. It has therefore been assumed, based on the results in Figure C.1, that the routine developed will provide an accurate enough correlation to that of GSP.



(a) AirTerm.m error: All properties



(b) AirTerm.m error: Excluding Entropy



(c) AirTerm.m error: Excluding Entropy and Specific Heats Ratio

Figure C.1: Comparison of air thermodynamic properties predicted by AirTherm.m routine and GSP

Appendix D

Turbo-Machine Performance Maps

Performance maps provide a convenient, yet sufficiently accurate, method of predicting compressor and turbine performance under different operating conditions. These maps provide information about the relationships between pressure ratio, rotational speed, mass flow rate and efficiency. The rotational speed and mass flow rate are usually expressed in "corrected" form described by Equations D.1 and D.2 respectively [33].

$$N_C = \frac{N}{\sqrt{\theta_{t,i}}} \tag{D.1}$$

$$\dot{m}_C = \dot{m}\frac{\sqrt{\theta_{t,i}}}{\delta_{t,i}} \tag{D.2}$$

where, $\theta_{t,i}$ and $\delta_{t,i}$ are the correction terms expressing the inlet total temperature and pressure, relative to standard conditions, respectively:

$$\theta_{t,i} = \frac{T_{t,i}}{T_{STP}} \tag{D.3}$$

$$\delta_{t,i} = \frac{P_{t,i}}{P_{STP}} \tag{D.4}$$

Corrected representations are obtained through dimensional analysis, accounting for Mach and Reynolds numbers [82], and allow for the valid use of these maps for varying flight conditions. Alternatively, as is the case in this work, it is common practice to represent corrected rotational speed as a percentage of the component's design corrected speed:

$$N_{C,\%design} = \frac{N_C}{N_{C,design}} \tag{D.5}$$

The parameters represented in these maps, are required to thermodynamically calculate the turbo-machine's off-design, steady-state performance, which is discussed in Sections E.3 and E.6 for compressors and turbines, respectively. These off-design, steady-state calculations may then be used in engine transient calculations

These performance maps, make the assumption of uniform inlet and exit flow. In addition, they are obtained by averaging the radial profiles of the flow properties. This is a reasonable assumption which produces reasonable results. Other types of performance maps exist, and are discussed in more detail in references [70–73, 75].

D.1 Compressor Performance Maps

Figure D.1 is an example of a compressor map, regenerated from GSP's BIGFAN sample model [113]. The BIGFAN sample model is used to simulate a typical, high-bypass, twin-spool, commercial turbofan engine and is based on General Electrics CF6-80 family. This particular example, is a generalised map used for representing the flow which passes through the core of the fan. In a typical compressor map, as with Figure D.1, pressure ratio is given as a function of corrected inlet mass flow rate, and is presented in the form of corrected rotational speed and efficiency isolines. In some cases, however, the efficiency is presented as a separate map, and is given as a function of corrected inlet mass flow rate using corrected speed isolines. Nonetheless, this work makes use of the former, more typical representation.

In Figure D.1 there is a relatively linear red line that terminates the speed lines at high PR values. Beyond this line the gradient of the speed line flattens out and sometime changes sign, which results in unstable operation of the compressor. Here, as the mass flow rate decreases so the PR drops, which in-turn causes the $\dot{m}_C$ to decrease further, thereby reducing the pressure ratio and so on. This unstable operation is known as surge, and hence the compressor is ideally confined to operate in the region of the map located below a surge line. The surge line is often off-set to provide a safety factor, this is because the actual point where surge occurs may shift due to external conditions. Compressor flow distortion is an example of operating conditions that may alter the effective position of the surge line on the compressor map [118]. Therefore, the surge line depicted in D.1 is positioned such that it intersects the speed isolines at a negative gradient. A high-pressure compressor map, on the other hand, will have a surge-line positioned a lot closer to the actual points where surge may occur. This is because the high pressure compressor is less likely to see distorted flow, which is corrected in the preceding compressor stages. Such and example is given by the HPC map used in this work; depicted in Figure D.9. In addition, as the speed isoline begins to flatten out, the change in mass flow rate

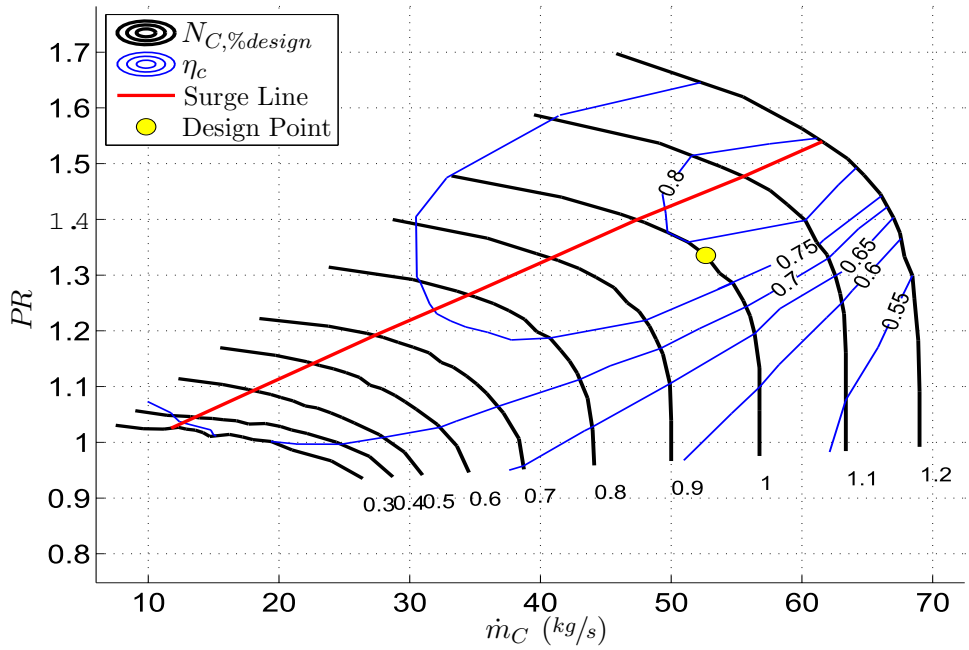


Figure D.1: Example of a compressor map, regenerated from GSP's BIGFAN sample model

becomes larger for a comparatively small change in pressure ratio. During transient operations, such large changes in mass flow rate for incremental changes in pressure ratio, can induce surging. This is another reason for the position of the surge line.

Compressor maps are usually generated from experimental data, and are kept proprietary by engine manufacturers. However, methods of map generalisation, preparation and scaling have been developed to modify existing maps to represent the performance of a desired compressor to be used to simulate a particular engine. This section describes some of those methods pertinent to this study.

D.1.1 Compressor Map Handling

Compressor maps are used to predict off-design compressor performance at different operating conditions. To do this computationally, maps may be represented as continuous functions or discretised to form look-up tables for inter/extrapolation. Both forms have their advantages and disadvantages, and are discussed here.

Continuous Representation of Compressor Maps

A popular method of modelling compressor maps is to use curve fits to represent speed isolines. Although analytical solutions are convenient and

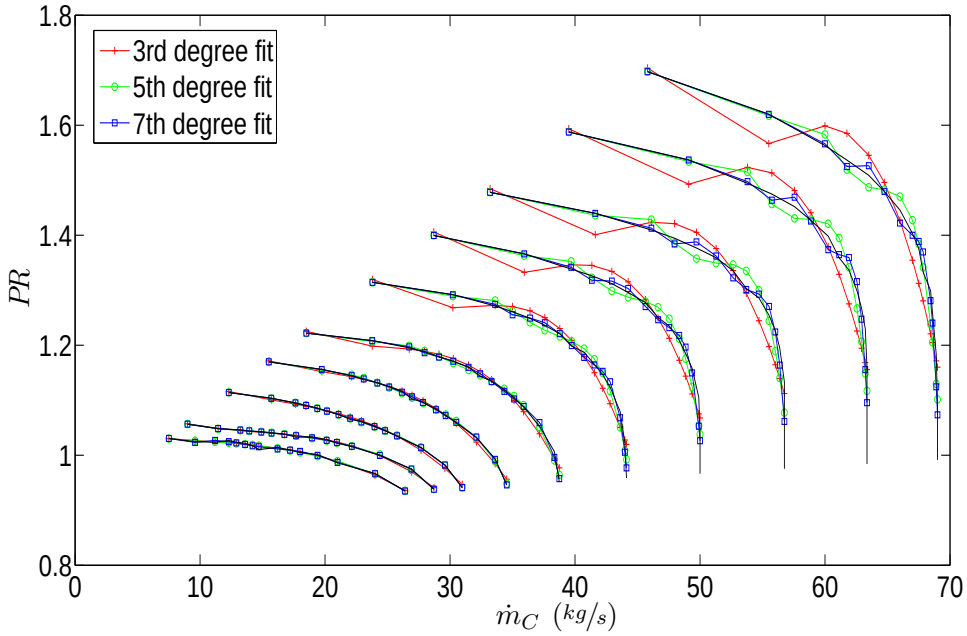


Figure D.2: Polynomial curve fits of compressor speed lines, applied to Figure D.1

easy to model, it is difficult to express compressor parameter relationships mathematically. Polynomial curve fits tend to be inaccurate because of oscillations about the actual speed line [119] as depicted in Figure D.2.

However, Moore and Greitzer [120] and Flack [33] have both proposed specialised polynomials, cubic and quadratic respectively, to approximate the shape of compressor map speed lines. Essentially the position of the inflection point or maxima, is manipulated to ensure intersection with the surge point. Other data points are then used to specify the rest of the function. Although these methods represent the physical relationships of compressor parameters quite nicely, they are just approximates and do not capture the true curvature of a compressors speed lines. Quadratics and cubic fits were applied to the map of Figure D.1, as depicted in Figure D.3, and gives an indication as to how these functions battle to conform to the curvature of the actual speed lines.

It is possible to fine tune polynomials fits, based on this idea, by varying which data points are used to define the coefficients of that function. Furthermore, polynomials of different orders may be used to better match curvature where required. Finally, the inflection point may be offset to better approximate the slope nearest the surge point. These variations may be used to arrange piecewise polynomials that effectively smooth speed line data. Figure D.4 is a simple example of how polynomials may be combined to best represent a speed line. It is obvious that the more polynomials used

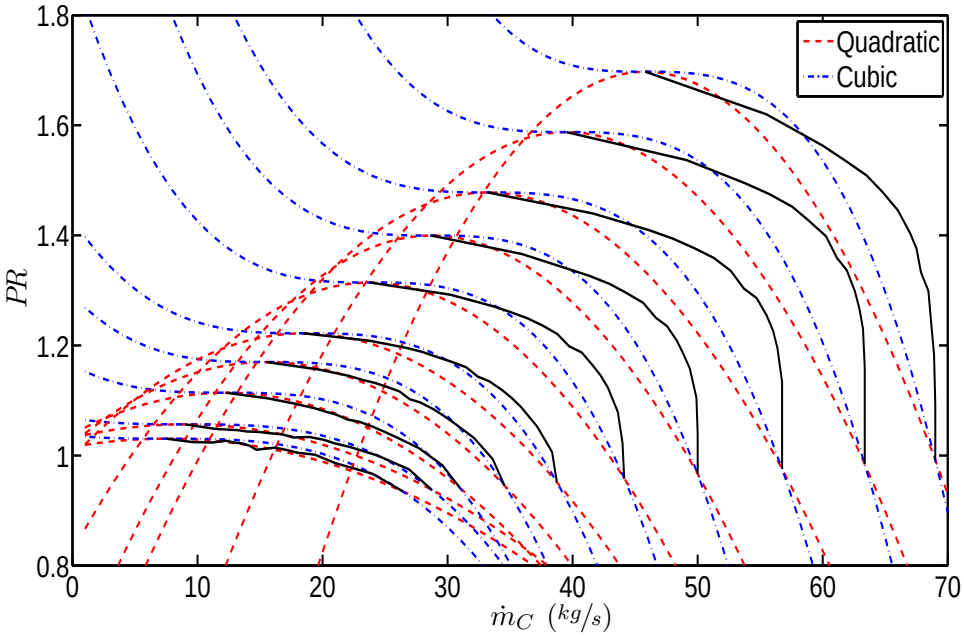


Figure D.3: Cubic and quadratic curve fits of compressor speed lines based on Moore Greitzer's [120] and Flack's [33] methods

per speed line, the better the approximation.

Drummond and Davison [119] have adapted Moore and Greitzer's approximation, through a similar tactic, by incorporating piece-wise splines into the smoothing process with improved results. However, these methods are unnecessarily complex and time consuming to set-up for the requirements of this work, due to the manual intensive smoothing required. This is because each speed line is of a unique shape, thereby making it difficult to automate the process. In addition, polynomial functions are smooth and therefore will not be able to conform to anomalies in the speed line, which are representative of the actual compressor's physics. Anomalies are evident in the very low speeds of Figure D.1, which show discontinuities within the speed line. These can of course be accountned for using linear segments, but further increases the complexity of the map's preparation for computational use. These methods may be appropriate for smoothing experimental compressor data, but there are simpler methods of computationally representing already prepared maps.

Discretised Representation of Compressor Maps

This work makes use of already prepared maps, obtained from GSP. Such compressor maps may be represented as discrete data points instead of continuous curves, which is convenient for computational use. The idea is to

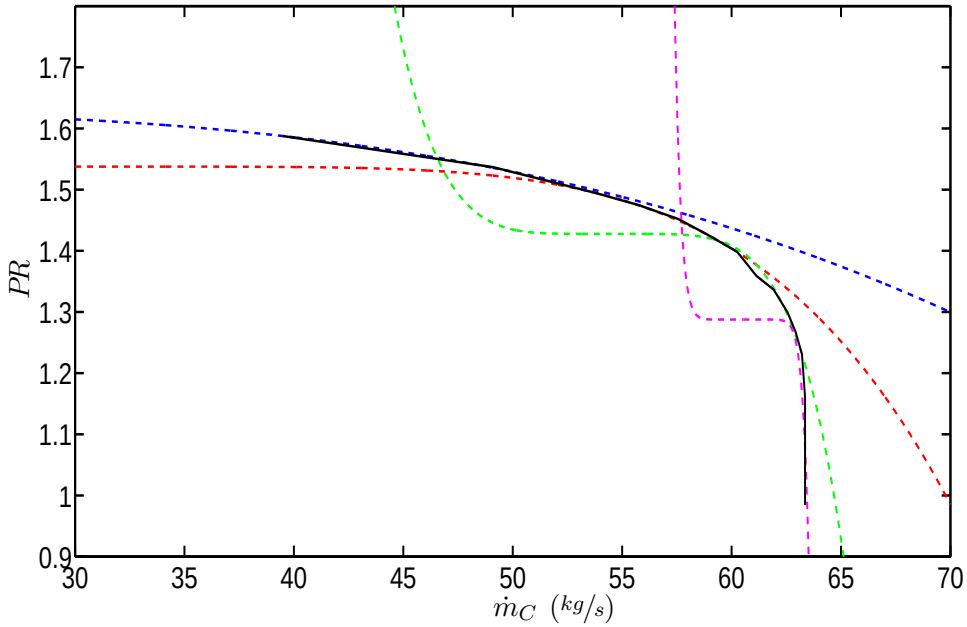


Figure D.4: Piece-wise polynomial curve fits of compressor map speed lines

generate look-up tables that describe the compressor's performance over its entire operating range. Such tables can be easily accessed and interpolated to determine compressor performance during engine simulations.

A particular map format is convenient when generating look-up tables. In this format, PR must be given as a function of corrected mass flow rate, and efficiency as a function of either pressure ratio or corrected mass flow rate. Furthermore, these relationships must be graphically represented as constant speed lines. Where this is not the case, available data points obtained experimentally or from an existing map must be smoothed, using one of the methods described earlier, for graphic representation. As an example, the efficiency contours in Figure D.1 give sparse information as to how efficiency varies along each speed line. This can be troublesome, as some speed lines intersect with only one efficiency contour. This is observed at low speeds in Figure D.1. It is advisable to smooth the speed lines relating efficiency and mass flow rate, individually and starting with those containing the most intersection points. In doing so, one may infer trends in the shapes of the speed lines which can then be used to help estimate reasonable curve fits of the remaining speed lines. Once completed, Figure D.1 can be graphically represented in the required format as shown in Figure D.5.

Once a map has been converted to the required format, it should be digitized appropriately for computational use. This is done by specifying discrete points along each speed line for both representations of PR and η_c as functions of $\dot{m}_C$. Unnecessary data points should be avoided for compu-

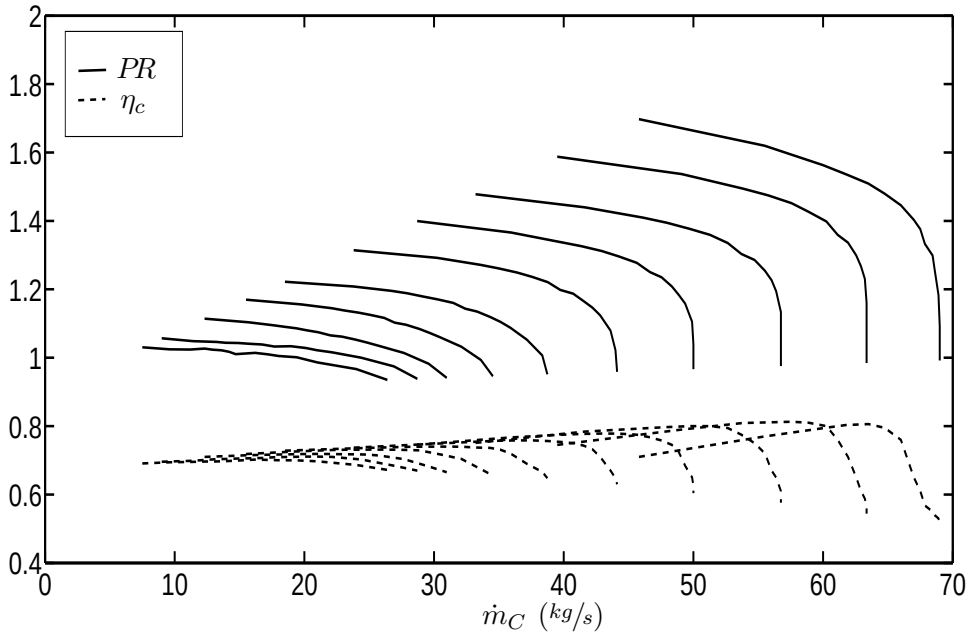


Figure D.5: Compressor map graphic format required for discretised map preparation

tational efficiency, and significant features (such as the surge point) should always be included. The digitised data may then be converted to look-up tables of an appropriate format for interpolation.

The digitised maps cannot simply be interpolated to predict compressor performance. This is because of possible occurrences of multiple solutions when interpolating. Kurzke [82] has developed unique methods of map preparation suited for use in complex engine simulations. Such methods are implemented using Smooth C software [104]. Here, look-up tables are generated from existing maps or experimental data, which may then be interpolated to predict compressor performance at various operating conditions. Kurzke's methodology, and some of the functionality of Smooth C, was replicated in the MATLAB environment to generate the look-up tables of the compressors to be modelled.

In this methodology, an auxiliary coordinate system is superimposed over the compressor's constant speed lines (In the PR versus $\dot{m}_C$ coordinate system). This auxiliary coordinate system is comprised of equispaced parabolic curves, each assigned a constant value ranging sequentially, and in equal increments, from $\beta = 0$ to $\beta = 1$. The parabolas are constructed such that they enclose, but do not overlap each other within the area of interest, and that the line representing $\beta = 1$ is the upper most curve which lies above the surge line and passes through the point $PR = 1$ where $\dot{m}_C = 0$ (if possible). Furthermore, for better physical representation, the parabola of

$N_C;\ \beta$	β_1	β_2	$\cdots$	β_n
$N_{C,1}$	$(PR, \dot{m}_C)_{1,1}$	$(PR, \dot{m}_C)_{1,2}$	$\cdots$	$(PR, \dot{m}_C)_{1,n}$
$N_{C,2}$	$(PR, \dot{m}_C)_{2,1}$	$(PR, \dot{m}_C)_{2,2}$	$\cdots$	$(PR, \dot{m}_C)_{2,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots\cdots$	$\vdots$
$N_{C,m}$	$(PR, \dot{m}_C)_{m,1}$	$(PR, \dot{m}_C)_{m,2}$	$\cdots$	$(PR, \dot{m}_C)_{m,n}$

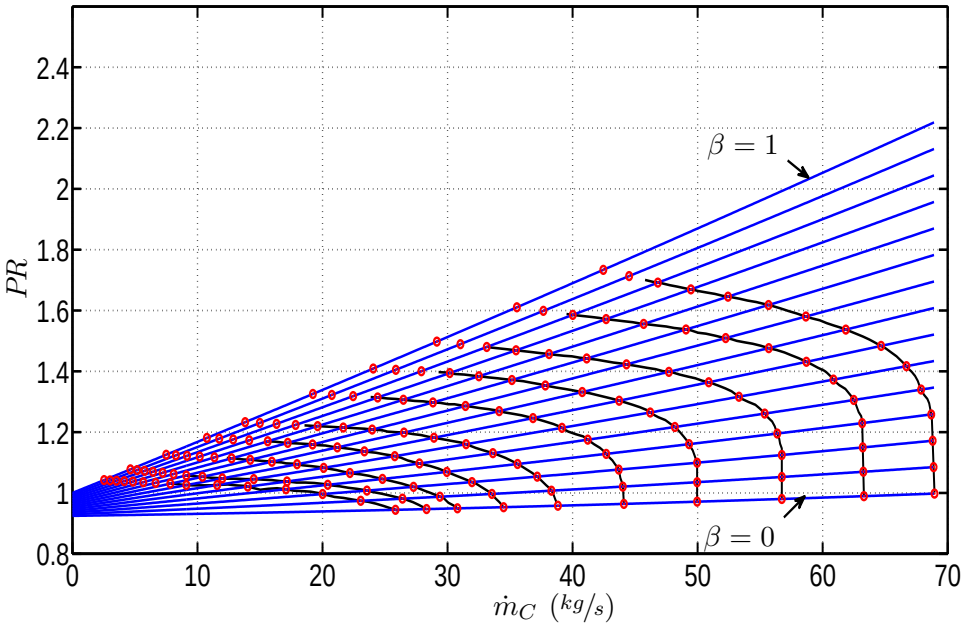
Table D.1: Format of compressor map look-up tables of size (n, m) respectively

$\beta = 0.5$ should coincide with the compressor's running line as closely as possible. The running line specifies compressors operating points at optimum efficiency across the steady-state operation of the overall engine. Therefore, when adapting a map for use in a different engine model, the running line must be determined from steady-state sweep calculations. If this is not possible, then the $\beta = 0.5$ line should correspond to the original maps running line for the best estimation. An automated β grid generator, based on Kurzke's work, was developed and applied to Figure D.1 to yield Figure D.6. Here, Figure D.6a depicts the interception points between PR speed lines (given as combinations of linear segments) and β curves. The efficiency points, interpolated from the digitised efficiency data, corresponding to these values are illustrated in Figure D.6b.

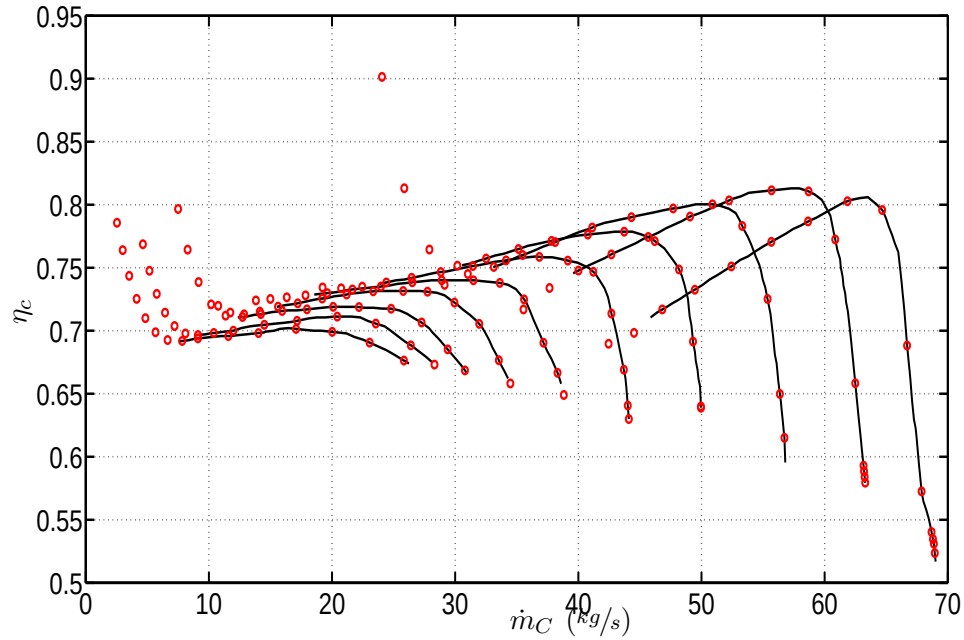
The density of the grid may be adjusted to increase accuracy, at a cost of computational efficiency. A dense β grid will generate large look-up tables which slows the interpolation procedure. Once the β grid has been refined, the intercepts are converted to three separate two-dimensional look-up tables containing $\dot{m}_C$, PR and η_c as functions of N_C and β. This format is in Table D.1.

Linear interpolation is then used to extract performance values at different operating conditions, given N_C and β. Kurzke recommends interpolating tables linearly because of speed and reliability in producing reasonable results. Furthermore, some maps contain discontinuities due to mechanical changes, such as variable inlet guide vanes, and linear interpolation will serve to better accommodate such discontinuities.

The advantage of preparing maps in this format is that, given a value of β and N_C, the corresponding values of η_c, $\dot{m}_C$ and PR remain distinct. Simple 2D linear interpolation of maps, without the β coordinate system, may encounter convergence issues because of the possibility of non-distinct values. This commonly occurs at high mass flow rates for a given speed, where speed lines become vertical, or at high pressure ratios where there exists local maxima.



(a) Pressure ratio speed lines with β grid and newly assigned data points



(b) Efficiency speed lines with corresponding newly assigned data points

Figure D.6: The generation of compressor map look-up tables

Smooth C [104] is commercial software that may be used to prepare maps in this way. Smooth C has the added advantage of being able to check maps for physical representativeness. This is done by analysing relationships of corrected flow and specific work, for each β value, as a function of corrected speed. Once the tabulated data has been correctly analysed, it is ready to be used in off-design and transient engine models to predict compressor performance. Lazzaretto [121] present a nice demonstration of Kurzke's method used to validate an artificial neural network model that predicts off-design performance of an single-spool engine with variable inlet guide vane (VIGV) load control. GSP also implements this method for the preparation of generalised maps to be used in its performance calculations.

D.1.2 Interpolation of Compressor Maps

The look-up tables generated by Kurzke's method aid the interpolation of compressor maps by providing distinct parameter values for a given corrected speed and β value. However, in order to use these tables in a dynamic simulation, one would have to include β as one of the parameters in the engine's state vector. This is most conveniently done when approaching engine dynamic simulations using the Iterative Method. However, when volume dynamics are required, it is difficult to translate volume pressure (or mass flow rate) derivatives to changes in β. Therefore, it is required that, to include volume dynamics, the compressors performance need be defined by combinations of either PR or $\dot{m}_C$ and N_C. This section describes possible approaches, still making use of the β coordinate system, to overcome non-distinct anomalies during interpolation of compressor maps. It is noted that the methods presented here are unique to this work, as far as the author is aware.

Using N_C and PR

During engine simulations, which include volume dynamics, the performance of the compressors are determined from maps where N_C and PR are given. The β auxiliary coordinate system is used to help select the appropriate operating point along a compressor map where the required PR may be non-distinct.

The parabolic β curves are roughly parallel, and similar in shape, to the compressor map's running line. This gives β curves physical significance in that they roughly resemble a path along which the compressor would, ideally, move from one steady-state operating point to the next. On this premise, the previous β value can be used to provide a crude estimate of the compressors operating position on the newly interpolated speed line. The gradient, of the new speed line, on either side of this estimated position is then established and compared against the difference between estimated and

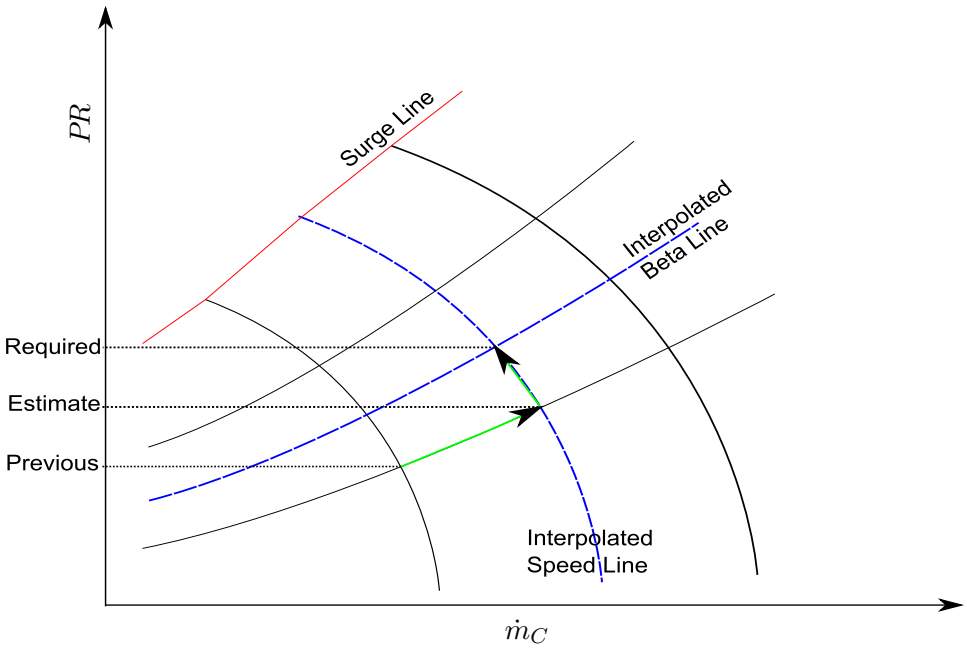


Figure D.7: Compressor map interpolation path given PR and N_C

required PR values. This comparison determines in which direction, along the new speed line, the interpolation procedure should search for a possible solution.

For example, if the integrated PR, from volume dynamics, is higher than the estimate value, and the most positive gradient is located to the left of the estimation point, then the search will initially traverse in the direction of decreasing $\dot{m}_C$ and converge on the first solution that it encounters. This is shown in Figure D.7.

If no solution is found along the initial choice in direction, the interpolation procedure will continue the search from the same initial estimation point but in the opposite direction. This may occur if the speed line has local maxima at $\dot{m}_C$ values larger than that of the surge point. Such local maxima may occur at low speeds or because of mechanical variations (e.g. variable inlet guide vanes). If the interpolation procedure still fails to converge, then no solution exists within bounds of the map and extrapolation is required. An simple extrapolation procedure has been included, as it is assumed that the acceleration/deceleration of the aircraft in the formation will not be so severe to cause the operation of compressors to fall outside their map bounds. However, it should be noted that should extrapolation be required, that this may introduce significant errors.

This interpolation procedure may also encounter errors when tracking transient shifts of compressor performance from one operating point to the next. This is because acceleration/deceleration of the compressor results in

a shift that deviates from the running line until the engine reaches steady-state operation once again. Such errors will only occur where solutions to a particular pressure ratio are non-distinct. Figure D.8a demonstrates how an anomaly in speed line shape, perhaps as a result of variable inlet guide vanes, could result in an interpolation error. These errors may be reduced by decreasing the time step of the simulation. In doing so, the original interpolated speed line is approached incrementally as indicated in Figure D.8b.

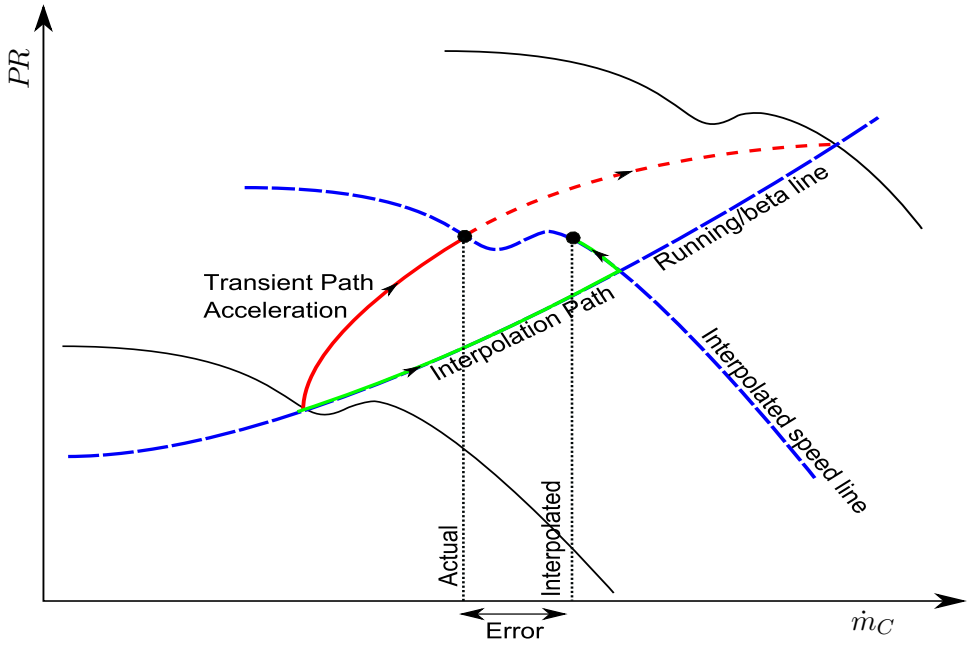
This also has the added benefit of tracking the actual transient shift more closely, but will slow down the overall simulation. However, anomalies such as these are normally found at low rotational speeds and therefore their effect on this study will be limited because of the high operational speeds at which engines run during cruise conditions. The procedures developed to model this approach has been detailed in Section D.1.4.

Using N_C and $\dot{m}_C$

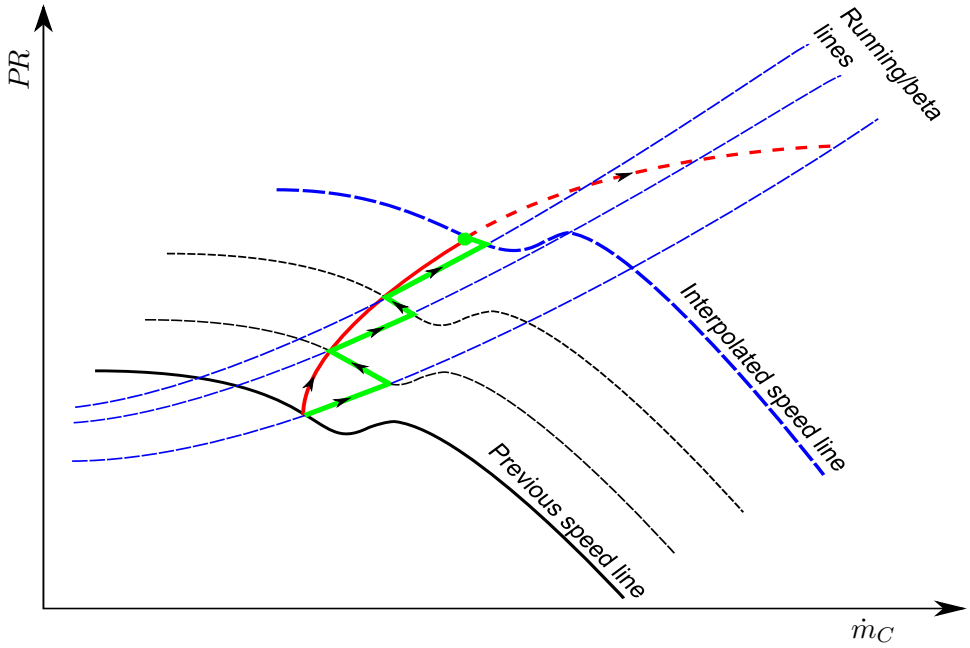
Depending on the method used for modelling volume dynamics, the compressor operation may be defined by the corrected inlet mass flow right and speed. In this case, the flow rate at a particular speed may reach maximum where the speed line becomes vertical (this is prominent at high speeds), making simple 2D interpolation difficult. To overcome this, it is assumed that if the required mass flow is not distinct then the interpolation will converge on the operating point of highest efficiency along the speed line, for that particular flow rate. This assumption is reasonable because it is very unlikely that the required flow rate will fall exactly on the vertical value of the speed line. If it does, any error will be corrected immediately within the next time step of a transient simulation, unless that next value also happens to coincide with the vertical portion of the next speed line. However, it is noted that this particular method is not used in the dynamic simulation method presented in this work. This work uses pressure ratio as one of the state variables defining the engine's dynamic operation. Therefore, this method of interpolation is not discussed further.

D.1.3 Compressor Map Scaling

When the component maps of the engine to be simulated are not available they may be scaled from other generalised maps, or maps from other engines. The simplest method is to linearly scale each map parameter by a ratio of the required to original map's design-point [61]. Equation D.6 is used for scaling all parameters (substituting X with the parameter of interest; N_C, $\dot{m}_C$ or η) except for pressure ratio which uses Equation D.7. A design point calculation of the entire engine is necessary to determine the design values corresponding to the engine to be modelled..



(a) Possible error scenario when interpolating from PR and N_C due to an anomaly in the speed line shape.



(b) Error reduction of Figure D.8a by reducing the simulation time step. The interpolation path zig-zags to track the actual transient path.

Figure D.8: Tracking transient paths across compressor map using interpolation procedure based on PR and N_C values

$$X = \left[\frac{X_{design}}{X_{map,design}}\right] X_{map} \tag{D.6}$$

$$PR = \left[\frac{PR_{design} - 1}{PR_{map,design} - 1}\right] (PR_{map} - 1) + 1 \tag{D.7}$$

This method may produce significant errors, even when scaling between components that are similar in type and operation [122]. This is because linear scaling does not necessarily account for the possible changes in map shape required to represent, physically, the off-design performance of the component.

Other scaling methods, that are more physically representative, may be used. One of the first methods uses computational parametric analysis to improve upon the afore mentioned scaling procedure [122]. Another method, gaining popularity, trains artificial neural-networks to generate maps based on existing examples from literature [123, 124]. More recently, the shape variance between available maps has been used to morphologically generate new maps according to a required design point [125].

However, one of the most commonly used and accepted scaling methods is that which is based on topological trends obtained through statistical analysis of the existing maps found in literature [126]. This method is implemented in the compressor and turbine map handling software Smooth C and T [104, 127], which is in-turn used to prepare maps for use in other commercial software packages such as GasTurb.

Nevertheless, despite the possible inaccuracies, this work implements Equations D.6 and D.7 because of its simplicity as well as falling seemingly in-line with the methods used in GSP. GSP does not explicitly detail its scaling methods, but the afore mentioned scaling method is eluded to in the User Manual [113].

D.1.4 MATLAB Routines

This subsection describes the routines developed for modelling the compressor maps. This portion of the model was programmed in MATLAB and outside of the Simulink environment. The code of the compressor map modelling, most relevant to the engine model, is presented here. Flow charts are presented to better demonstrate the procedures uses, where deemed necessary.

Compressor Maps Library

Included in this model is a library of four generalised maps, obtained from GSP's BIGFAN sample model, to be used when modelling a typical high bypass turbofan engine. These include a high, intermediate and two low

pressure compressor maps (for modelling the fan's core and bypass flows), which are depicted in Figures D.9, D.10, D.11 and D.12, respectively.

These maps were each discretised to obtain tables of the format depicted in Table D.1. These tables are stored in an m-file, as demonstrated in Code Listing D.1. The actual table entries have been omitted from this code listing to save space in the document. In addition, only the supporting code and comments for one map has been included, as the remaining code is just a repeat of that entry. Therefore, this architecture allows for easy expansion of the compressor map library. As demonstrated in Code Listing D.1, vectors are required for β and N_C values. Thereafter, corresponding values of $\dot{m}_C$, PR and η_c are tabulated in arrays where the each row represents a constant speed line and each column a constant β line. The surge line is represented as a single array containing both PR and $\dot{m}_C$ values corresponding to its position on each constant speed line. Each column represents a speed line with its first and second rows containing the corresponding values of $\dot{m}_C$ and PR, respectively. Finally, the map's design point is defined by a given coordinate of $N_{C,design}$ and β_{design}, from which the other values of PR_{design}, $\dot{m}_{C,design}$ and η_{design} are interpolated from the map. Because both N_C and β have been defined, there is no issue in encountering a scenario where values are non-distinct during interpolation. Therefore, the library takes in an input of which type of map to be used, and returns the tables as well as the map's design point.

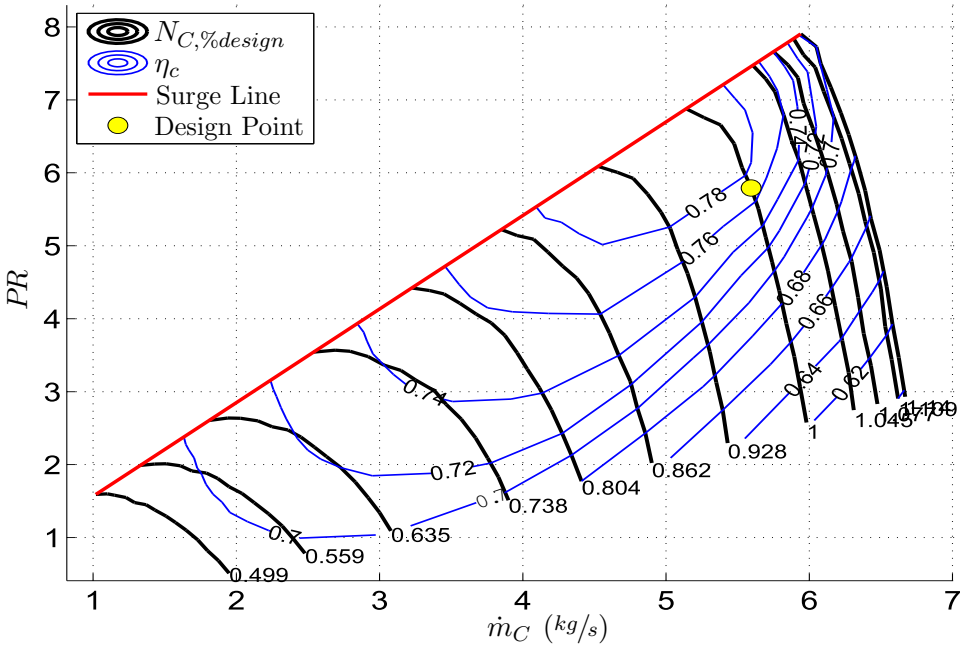


Figure D.9: High pressure compressor map

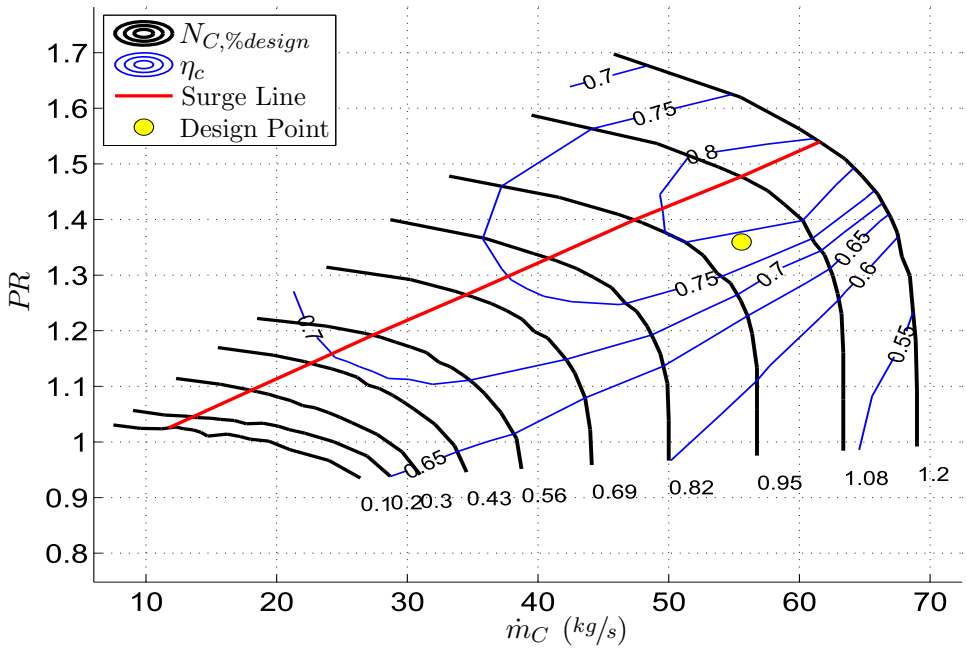


Figure D.10: Intermediate pressure compressor map

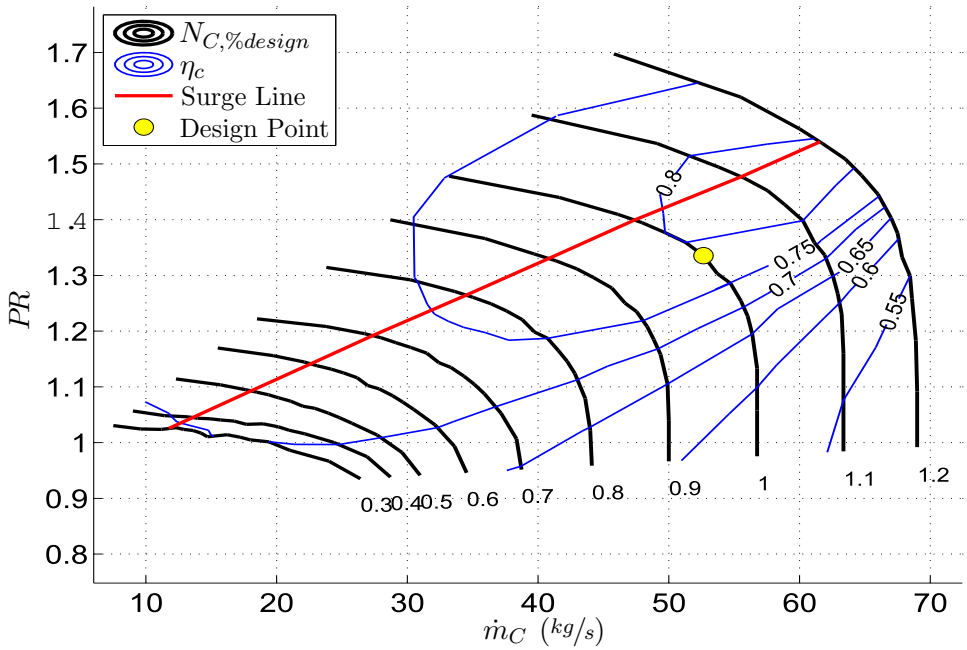


Figure D.11: Low pressure compressor map (core)

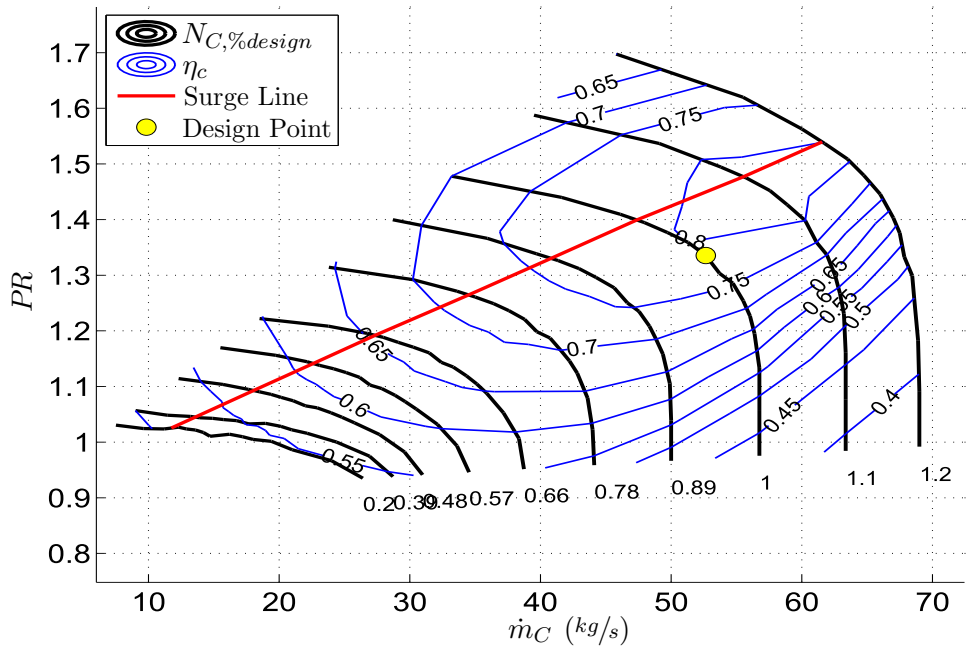


Figure D.12: Low pressure compressor map (bypass)

Code Listing D.1: MATLAB Function CompMaps.m: Library of compressor maps

```
%=================================================================
%                       Compressor Map Data
%=================================================================
%
% NAME:             CompMaps.m
%
% AUTHOR:           D. Sanders, University of Cape Town
%
% DATE:             12 October 2012
%
% FUNCTION:         [betaVect, speeVect, massFlowTabl, ...
%                   presRatiTabl, effiTabl, surgVect, mapDP]...
%                   = CompMaps (mapType)
%
% PURPOSE:          Provide data representing generalised
%                   compressor maps which may be scaled
%                   according to engine design point values
%
% INPUTS:
%
%   mapType         = String use to select appropriate map
%                   ('LPCD' 'LPCC' 'IPC' or 'HPC')
%
% OUTPUTS:
```

```
25 %
26 %   betaVect        = Vector of beta grid values           [-]
27 %   speeVect        = Vector of iso speed lines     [%Ndesign]
28 %   massFlowTabl    = Corrected mass flow table         [kg/s]
29 %   presRatiTabl    = Pressure ratio table                 [-]
30 %   effiTabl        = Efficiency table                 [%/100]
31 %   surgVect        = Surge coordinates per speed line
32 %
33 % SUBROUTINES:      None
34 %
35 % REFERENCES:
36 %
37 %   Data Obtained from:
38 %       GSP BIGFAN example model. NLR, Gas Turbine Simulation
39 %       Program 2011.
40 %
41 % For coding conventions, please refer to:
42 %
43 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
44 %   ver 1.5. October 2002.
45 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
46 %=============================================================
47
48 function [betaVect, speeVect, massFlowTabl, presRatiTabl, ...
49     effiTabl, surgVect, mapDP] = CompMaps (mapType)
50
51 % Select generalised map to be used
52 switch mapType
53
54 %-------------------------------------------------------------
55 % LPCD - Generalised low pressure compressor map (bypass)
56 %-------------------------------------------------------------
57
58 case {'LPCD'}
59
60 % Beta Refernce Grid Values (row vector)
61 betaVect = [...];
62
63 % Constant Speed Lines (row vector)
64 speeVect = [...];
65
66 % Mass Flow Table (Rows,Cols)=(Constant Speed, Constant Beta)
67 massFlowTabl = [...];
68
69 % Efficiency Table (Rows,Cols)=(Constant Speed, Constant Beta)
70 effiTabl = [...];
71
72 % Press Ratio Table (Rows,Cols)=(Constant Speed, Constant Beta)
73 presRatiTabl = [...];
74
75 % Surge Line (Rows,Cols)=([Mass Flow; Pressure Ratio], Speed)
76 surgVect = [...];
77
78 % Specify the design point of the unscaled map
```

```
79 % mapDP = [beta Nc Wc ETA PR];
80
81 Nd = ...;
82 BETAd = ...;
83 PRd  = interp2(speeVect, betaVect, presRatiTabl', Nd, BETAd);
84 WCd  = interp2(speeVect, betaVect, massFlowTabl', Nd, BETAd);
85 ETAd = interp2(speeVect, betaVect, effiTabl'    , Nd, BETAd);
86
87 mapDP = [BETAd Nd WCd ETAd PRd];
88
89 %================================================================
90 % Repeated for high, intermediate and low (core) pressure maps
```

Compressor Map Interpolation

The interpolation procedure is depicted by the flow charts of Figures D.13 and D.14. In Figure D.13, the procedure begins by selecting the type of generalised map to be used. This is because there are large differences between the characteristics of a high, medium and low pressure compressors. The map's tables are imported from a library, as described in the previous subsection using Code Listing D.1, and scaled according to the design point specified for that particular component (see Equations D.6 and D.7).

The operating speed of the compressor is then checked to see whether it falls within the bounds of the compressor map. If this is not so, a warning is issued to the command line (stating how the operating point falls outside of the map's bounds) and a simple extrapolation procedure (MATLAB's built-in interpolation function) is used to approximate the compressor's operating point. If the compressor speed does fall within the limits of the map, a new speed line may be interpolated from the compressor map tables based on that speed's value.

It is then necessary to check as to whether the prescribed pressure ratio is located within the bounds of the newly interpolated speed line. If this is not so, then similarly a warning is issued and the speed line is extrapolated. Alternatively, if the pressure ratio is found within the limits of the speed line, then the interpolation procedure may continue. It is noted that any warnings issued will specify details as to how the map's bounds were exceeded. This is important for diagnosing issues with either the model or its inputs (for example, this may occur when using the wrong map type to model a compressor).

Upon confirming that the compressor's operating speed and pressure ratio fall within the map's bounds, an estimation of its new operating point may be interpolated on the new speed line using the previous β value. The resulting and specified pressure ratios are then compared to check as to whether this estimation is correct. If this estimation is correct, then the new compressor operating point is assigned those values corresponding to

the estimation. These values are then compared to the surge line, and a warning is issued if surge conditions are met. However, it is very unlikely that this estimation will correctly correlate to the specified pressure ratio. In this event, the routine will traverse the speed line in search of a solution. This traversal is detailed in Figure D.14.

The traversal procedure, first determines the speed line's gradients directly adjacent to the estimation point. These gradients, along with a comparison of the estimated and specified pressure ratio, are used to determine the direction along the speed line in which a solution is most likely to occur soonest. This is thought to be physically representative of the actual compressors operation as it is intuitive that the compressor would follow a path of least resistance. Figure D.14 illustrates how the routine sequentially checks the discretised points along the speed line, for an interval in-which the specified pressure ratio will lie. Once an interval has been found, the correct operating point may be interpolated according to the values corresponding to that interval. It is noted that the appropriate checks have been made to ensure that a solution does exist within the chosen direction of traversal, thereby mitigating any concerns of the search exceeding the bounds of the speed line in that chosen direction.

The reader is once again referred to Section D.1.2, and in particular to Figure D.7, for a better understanding as to how this interpolation routine is intended to function.

The interpolation routine is presented in Code Listing D.2 and it is left to the reader to interpret this code with the assistance of Figures D.13 and D.14 as well as the explanations given thus far. The code is also well commented to aid in its explanation.

Thus far, the methodology for storing, scaling and interpolating compressor maps has been discussed. These methods are validated against GSP in the following Subsection.

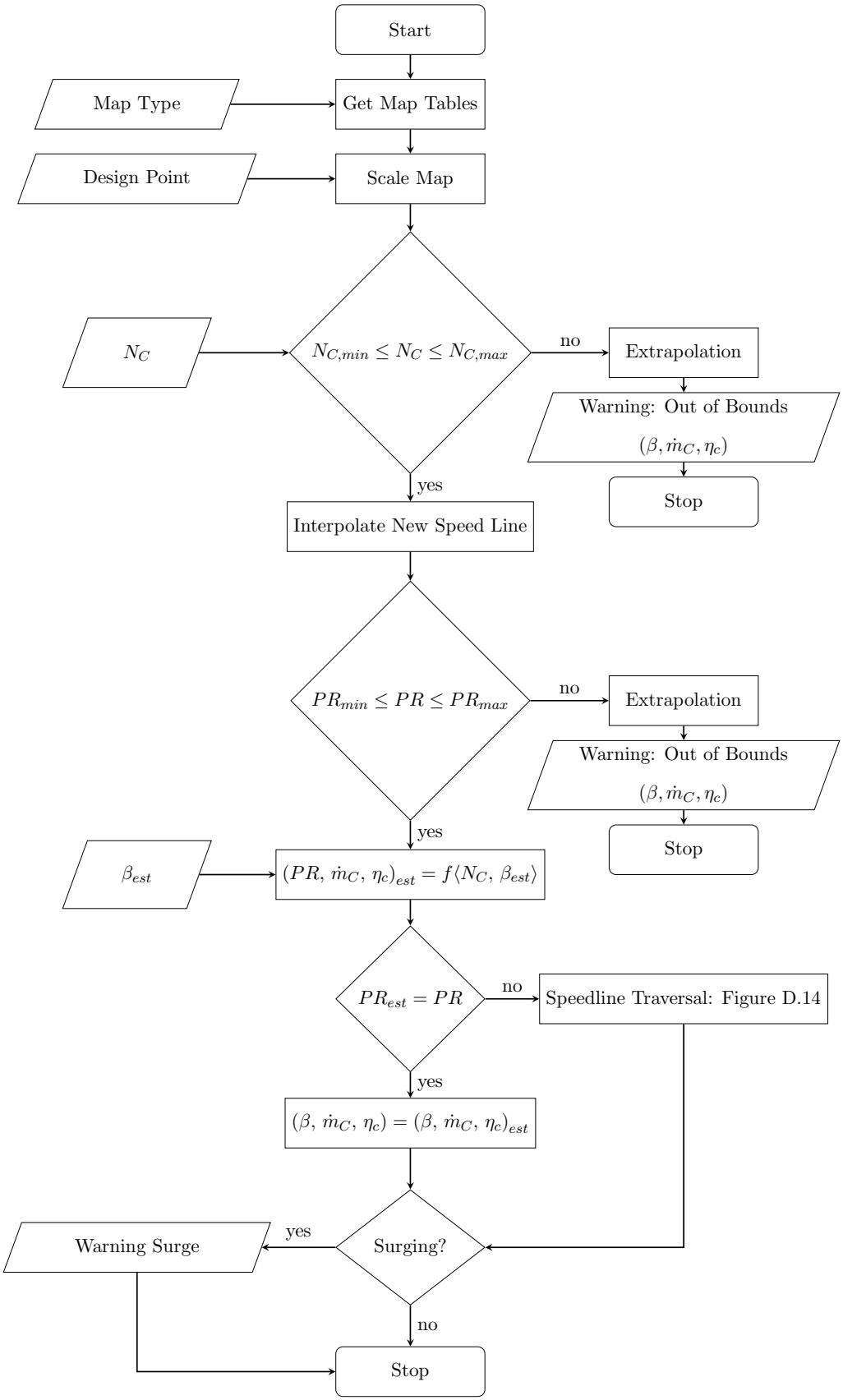


Figure D.13: Flow chart: Compressor map interpolation - Flow Chart 1

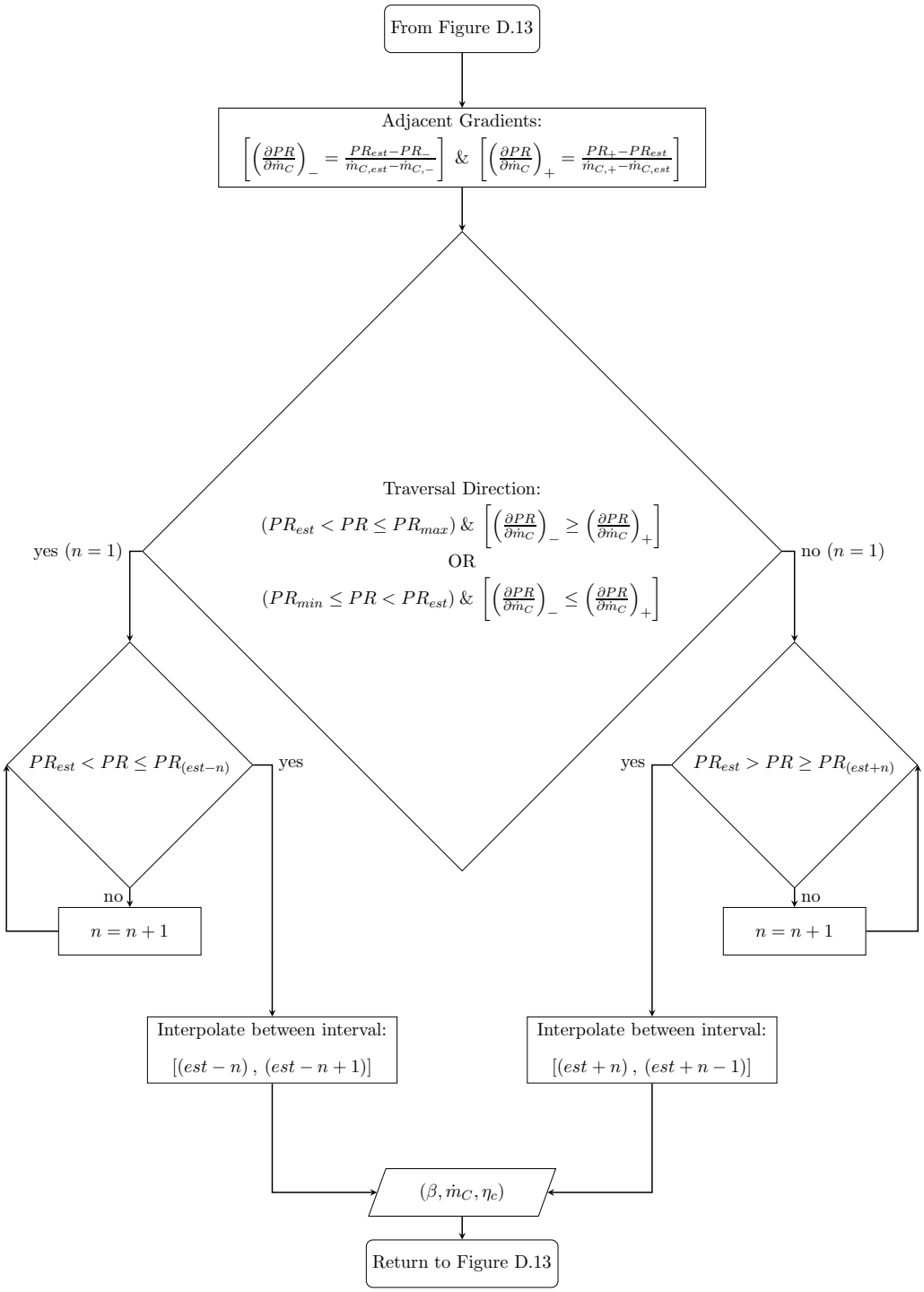


Figure D.14: Flow chart: Compressor map interpolation - Flow Chart 1

Code Listing D.2: MATLAB Function CompMapsInterpPR.m: For interpolating compressor maps given N_C and PR

```
%============================================================
%               Compressor Map Interpolation Scheme
%============================================================
%
% NAME:             CompMapsInterpPR.m
%
% AUTHOR:           D. Sanders, University of Cape Town
%
% DATE:             12 October 2012
%
% FUNCTION:         [BETA massFlowValu effiValu] = ...
%                  CompMapsInterpPR (N, PR, BETAi, mapType, DP)
%
%
% PURPOSE:          Interpolate compressor off-design
%                   operating parameters given the pressure
%                   ratio
%
% INPUTS:
%
%   N               = Corrected Spool Speed              [rpm]
%   PR              = Pressure Ratio                       [-]
%   BETAi           = Initial beta value estimate          [-]
%   mapType         = Compressor map type             [string]
%   DP              = Design point parameters of compressor
%
% OUTPUTS:
%
%   BETA            = Beta grid interpolated value         [-]
%   massFlowValu    = Corrected mass flow rate          [kg/s]
%   effiValu        = Efficiency                       [%/100]
%
% SUBROUTINES:      CompMaps.m
%
% REFERENCES:
%
%   Scaling method from:
%       Sellers, J. & Daniele, C.
%       "DYNGEN: A program for calculating steady-state and
%       transient performance of turbojet and turbofan
%       engines", 1975
%
% For coding conventions, please refer to:
%
%   Johnson, Richard. "MATLAB Programming Style Guidlines".
%   ver 1.5. October 2002.
%   http://www.mathworks.com/matlabcentral/fileexchange/2529
%============================================================
%% Compressor Map Off Design operation Predictor

function    [BETA massFlowValu effiValu] = ...
```

```
52              CompMapsInterpPR (N, PR, BETAi, mapType, DP)
53
54  %% Map Preparation
55
56  % Fetch appropriate map data
57  [betaVect, speeVect, massFlowTabl, presRatiTabl, effiTabl, ...
        surgVect, mapDP] = CompMaps (mapType);
58
59  if size(DP) ≠ size(mapDP)
60      DP = DP';
61  end
62
63  % Calculate Map Scaling Factors
64  SF = DP ./ mapDP;
65  SF(5) = (DP(5) - 1) / (mapDP(5) - 1);
66
67  % SF = [SFbeta SFnc SFwc SFeta SFpr]
68  betaVect         = SF(1) * betaVect;
69  speeVect         = SF(2) * speeVect; % Changes %-->rpm
70  massFlowTabl     = SF(3) * massFlowTabl;
71  effiTabl         = SF(4) * effiTabl;
72  presRatiTabl     =(SF(5) * presRatiTabl) - SF(5) + 1;
73
74  % Must also scale surge line values
75  surgVect(1,:) = SF(3) * surgVect(1,:);
76  surgVect(2,:) =(SF(5) * surgVect(2,:)) - SF(5) + 1;
77
78  %% Find Surge Point for Speed
79
80  % Mass Flow & Pressure Ratio
81  surgPoin = [   interp1(speeVect, surgVect(1,:), N)
82                 interp1(speeVect, surgVect(2,:), N)];
83
84  %% Interpolate Off-Design Compressor Performance
85
86  % Simple interpolation procedure used for comparison
87  %=============================================================
88  %     PRsline = interp1(speeVect, presRatiTabl, N, ...
        'linear', 'extrap');
89  %     WCsline = interp1(speeVect, massFlowTabl, N,  ...
        'linear', 'extrap');
90  %     ETAsline = interp1(speeVect, effiTabl, N,  'linear', ...
        'extrap');
91  %
92  %     BETA = interp1(PRsline, betaVect, PR, 'linear', ...
        'extrap');
93  %     massFlowValu = interp1(PRsline, WCsline, PR, ...
        'linear', 'extrap');
94  %     effiValu = interp1(PRsline, ETAsline, PR, 'linear', ...
        'extrap');
95
96  % Step 1: Parameters along interpolated speedline
97  %=============================================================
98
```

```
99  % Check if Spool Speed Falls within in Map Bounds
100 if N > max(speeVect)
101
102     PRsline = interp1(speeVect, presRatiTabl, N, 'linear', ...
            'extrap');
103     WCsline = interp1(speeVect, massFlowTabl, N,  'linear', ...
            'extrap');
104     ETAsline = interp1(speeVect, effiTabl, N,  'linear', ...
            'extrap');
105
106     BETA = interp1(PRsline, betaVect, PR, 'linear', 'extrap');
107     massFlowValu = interp1(PRsline, WCsline, PR, 'linear', ...
            'extrap');
108     effiValu = interp1(PRsline, ETAsline, PR, 'linear', ...
            'extrap');
109
110     if strcmp(mapType, 'LPCC') == 1
111     warning('LPCC: Interpolation out of compressor map ...
            bounds: N too high')
112
113     elseif strcmp(mapType, 'LPCD') == 1
114     warning('LPCD: Interpolation out of compressor map ...
            bounds: N too high')
115
116     elseif strcmp(mapType, 'IPC') == 1
117     warning('IPC: Interpolation out of compressor map ...
            bounds: N too high')
118
119     elseif strcmp(mapType, 'HPC') == 1
120     warning('HPC: Interpolation out of compressor map ...
            bounds: N too high')
121
122     end
123
124 elseif N < min(speeVect)
125
126     PRsline = interp1(speeVect, presRatiTabl, N, 'linear', ...
            'extrap');
127     WCsline = interp1(speeVect, massFlowTabl, N,  'linear', ...
            'extrap');
128     ETAsline = interp1(speeVect, effiTabl, N,  'linear', ...
            'extrap');
129
130     BETA = interp1(PRsline, betaVect, PR, 'linear', 'extrap');
131     massFlowValu = interp1(PRsline, WCsline, PR, 'linear', ...
            'extrap');
132     effiValu = interp1(PRsline, ETAsline, PR, 'linear', ...
            'extrap');
133
134     if strcmp(mapType, 'LPCC') == 1
135     warning('LPCC: Interpolation out of compressor map ...
            bounds: N too low')
136
137     elseif strcmp(mapType, 'LPCD') == 1
```

```
138     warning('LPCD: Interpolation out of compressor map ...
            bounds: N too low')
139
140     elseif strcmp(mapType, 'IPC') == 1
141     warning('IPC: Interpolation out of compressor map ...
            bounds: N too low')
142
143     elseif strcmp(mapType, 'HPC') == 1
144     warning('HPC: Interpolation out of compressor map ...
            bounds: N too low')
145
146     end
147 else
148     PRsline = interp1(speeVect, presRatiTabl, N);
149     WCsline = interp1(speeVect, massFlowTabl, N);
150     ETAsline = interp1(speeVect, effiTabl, N);
151
152 % Check if Pressure Ratio Falls within in Speed Line Bounds
153 if PR > max(PRsline)
154
155     BETA = interp1(PRsline, betaVect, PR, 'linear', 'extrap');
156     massFlowValu = interp1(PRsline, WCsline, PR, 'linear', ...
            'extrap');
157     effiValu = interp1(PRsline, ETAsline, PR, 'linear', ...
            'extrap');
158
159     if strcmp(mapType, 'LPCC') == 1
160     warning('LPCC: Interpolation out of compressor map ...
            bounds: PR too high')
161
162     elseif strcmp(mapType, 'LPCD') == 1
163     warning('LPCD: Interpolation out of compressor map ...
            bounds: PR too high')
164
165     elseif strcmp(mapType, 'IPC') == 1
166     warning('IPC: Interpolation out of compressor map ...
            bounds: PR too high')
167
168     elseif strcmp(mapType, 'HPC') == 1
169     warning('HPC: Interpolation out of compressor map ...
            bounds: PR too high')
170
171     end
172
173 elseif PR < min(PRsline)
174
175     BETA = interp1(PRsline, betaVect, PR, 'linear', 'extrap');
176     massFlowValu = interp1(PRsline, WCsline, PR, 'linear', ...
            'extrap');
177     effiValu = interp1(PRsline, ETAsline, PR, 'linear', ...
            'extrap');
178
179     if strcmp(mapType, 'LPCC') == 1
180     warning('LPCC: Interpolation out of compressor map ...
```

```
        bounds: PR too low')
181
182     elseif strcmp(mapType, 'LPCD') == 1
183     warning('LPCD: Interpolation out of compressor map ...
            bounds: PR too low')
184
185     elseif strcmp(mapType, 'IPC') == 1
186     warning('IPC: Interpolation out of compressor map ...
            bounds: PR too low')
187
188     elseif strcmp(mapType, 'HPC') == 1
189     warning('HPC: Interpolation out of compressor map ...
            bounds: PR too low')
190
191     end
192
193 else
194
195 % Check and correct BETA if it falls outside of its limits
196 if (BETAi > 1)
197     BETAi = 1;
198 elseif (BETAi < 0)
199     BETAi = 0;
200 end
201
202 % Step 2: Find Pressure Ratio at current BETA value
203 %============================================================
204 PRi     = interp2(speeVect, betaVect, presRatiTabl' , N, ...
        BETAi);
205 WCi     = interp2(speeVect, betaVect, massFlowTabl' , N, ...
        BETAi);
206 ETAi    = interp2(speeVect, betaVect, effiTabl'     , N, ...
        BETAi);
207
208 % Step 3: Check if PR is given by initial BETA estimate
209 %============================================================
210 if PRi == PR
211
212     BETA = BETAi;
213     massFlowValu = WCi;
214     effiValu = ETAi;
215
216     % Check for surging
217     if (massFlowValu ≤ surgPoin(1,1)) || (PR ≥ surgPoin(2,1))
218
219         if strcmp(mapType, 'LPCC') == 1
220         warning('LPCC: Map Interpolation successful but ...
                Compressor Surging!')
221
222         elseif strcmp(mapType, 'LPCD') == 1
223         warning('LPCD: Map Interpolation successful but ...
                Compressor Surging!')
224
225         elseif strcmp(mapType, 'IPC') == 1
```

```
226         warning('IPC: Map Interpolation successful but ...
                Compressor Surging!')
227
228         elseif strcmp(mapType, 'HPC') == 1
229         warning('HPC: Map Interpolation successful but ...
                Compressor Surging!')
230
231         end
232
233     end
234
235     return
236
237 end
238
239 % Step 3: Check initial BETA is within Speed line Bounds
240 %============================================================
241
242 % Minimum BETAi --> Move in direcion of decreasing mass flow
243 if BETAi == min(betaVect)
244
245     % Use standard 2d interpolation
246     BETA = interp1( PRsline, betaVect, PR);
247     massFlowValu  = interp2(speeVect, betaVect, ...
            massFlowTabl', N, BETA);
248     effiValu = interp2(speeVect, betaVect, effiTabl', N, BETA);
249
250     % Check for surging
251     if (massFlowValu ≤ surgPoin(1,1)) || (PR ≥ surgPoin(2,1))
252
253         if strcmp(mapType, 'LPCC') == 1
254         warning('LPCC: Map Interpolation successful but ...
                Compressor Surging!')
255
256         elseif strcmp(mapType, 'LPCD') == 1
257         warning('LPCD: Map Interpolation successful but ...
                Compressor Surging!')
258
259         elseif strcmp(mapType, 'IPC') == 1
260         warning('IPC: Map Interpolation successful but ...
                Compressor Surging!')
261
262         elseif strcmp(mapType, 'HPC') == 1
263         warning('HPC: Map Interpolation successful but ...
                Compressor Surging!')
264
265         end
266
267     end
268
269     return
270
271 % Maximum BETAi --> Move in direcion of increasing mass flow
272 elseif BETAi == max(betaVect)
```

```
273
274     % Use standard 2d interpolation
275     BETA = interp1( fliplr(PRsline), fliplr(betaVect), PR);
276     massFlowValu  = interp2(speeVect, betaVect, ...
            massFlowTabl', N, BETA);
277     effiValu = interp2(speeVect, betaVect, effiTabl', N, BETA);
278
279     % Check for surging
280     if (massFlowValu ≤ surgPoin(1,1)) || (PR ≥ surgPoin(2,1))
281
282         if strcmp(mapType, 'LPCC') == 1
283         warning('LPCC: Map Interpolation successful but ...
                Compressor Surging!')
284
285         elseif strcmp(mapType, 'LPCD') == 1
286         warning('LPCD: Map Interpolation successful but ...
                Compressor Surging!')
287
288         elseif strcmp(mapType, 'IPC') == 1
289         warning('IPC: Map Interpolation successful but ...
                Compressor Surging!')
290
291         elseif strcmp(mapType, 'HPC') == 1
292         warning('HPC: Map Interpolation successful but ...
                Compressor Surging!')
293
294         end
295
296     end
297
298     return
299
300 end
301
302 % Step 4: Find adjacent nodes and determine slope of speedline
303 %=============================================================
304 node1 = find(BETAi ≥ betaVect, 1, 'last');
305 node2 = find(BETAi ≤ betaVect, 1, 'first');
306
307 if node1 == node2
308     node1 = node1 - 1;
309     node2 = node2 + 1;
310 end
311
312 grad1 = (PRsline(node1) - PRi) / abs(WCsline(node1) - WCi);
313 grad2 = (PRsline(node2) - PRi) / abs(WCsline(node2) - WCi);
314
315 % Step 5: Locate possible solutions in both directions
316 %=============================================================
317
318 % Dir1 - direction 1 moves with decreasing corrected mass flow
319 % Dir2 - direction 1 moves with increasing corrected mass flow
320
321 % Set direction vectors for beta and pressure ratio
```

```
322 betaDir1 =  betaVect(node1:end);
323 betaDir2 =  fliplr( betaVect(1:node2) );
324
325 PRDir1 =  PRsline(node1:end);
326 PRDir2 =  fliplr( PRsline(1:node2) );
327
328 % Select if direction 1 or direction 2 should be searched first
329 if ((PR > PRi) && (grad2 ≥ grad1)) || ((PR < PRi) && (grad2 ...
    < grad1))
330     PRDirA = PRDir1; betaDirA = betaDir1;
331     PRDirB = PRDir2; betaDirB = betaDir2;
332 else
333     PRDirA = PRDir2; betaDirA = betaDir2;
334     PRDirB = PRDir1; betaDirB = betaDir1;
335 end
336
337 % Search initial direction choice
338     for iDirA = 1 : length(PRDirA) - 1
339
340     if PRDirA(iDirA) == PR
341         BETA = betaDirA(iDirA);
342
343     elseif PRDirA(iDirA + 1) == PR
344         BETA = betaDirA(iDirA+1);
345
346     else
347         logicDirA = logical( (PRDirA(iDirA) < PR && PR < ...
                PRDirA(iDirA+1))...
348                         || (PRDirA(iDirA) > PR && PR > ...
                            PRDirA(iDirA+1)) );
349
350         if logicDirA == 1
351             BETA = interp1( PRDirA(iDirA : iDirA+1), ...
352                             betaDirA(iDirA : iDirA+1), PR) ...
                                ;
353             break
354
355         else
356
357             BETA = -1;
358
359         end
360     end
361     end
362
363 % If beta = -1 then search needs to traverse in opposite ...
    direction
364 if BETA == -1
365
366     for iDirB = 1 : length(PRDirB) - 1
367
368     if PRDirB(iDirB) == PR
369         BETA = betaDirB(iDirB);
370
```

```
371     elseif PRDirB(iDirB + 1) == PR
372         BETA = betaDirB(iDirB+1);
373 
374     else
375         logicDirB = logical( (PRDirB(iDirB) < PR && PR < ...
                PRDirB(iDirB+1))...
376                         || (PRDirB(iDirB) > PR && PR > ...
                                PRDirB(iDirB+1)) );
377 
378         if logicDirB == 1
379             BETA = interp1( PRDirB(iDirB : iDirB+1), ...
380                             betaDirB(iDirB : iDirB+1), PR);
381 
382             % Step 6: Use new beta to find mass flow and ...
                    efficiency
383             %=============================================================
384             massFlowValu    = interp2(speeVect, betaVect, ...
                    massFlowTabl', N, BETA);
385             effiValu        = interp2(speeVect, betaVect, ...
                    effiTabl', N, BETA);
386 
387             break
388 
389         else
390 
391             BETA = interp1(PRsline, betaVect, PR, 'linear', ...
                    'extrap');
392             massFlowValu = interp1(PRsline, WCsline, PR, ...
                    'linear', 'extrap');
393             effiValu = interp1(PRsline, ETAsline, PR, ...
                    'linear', 'extrap');
394 
395             break
396 
397 %               if strcmp(mapType, 'LPCC') == 1
398 %               error('LPCC: Interpolation unsuccesful. ...
        Boundaries checked, solution should exist. check code')
399 %
400 %               elseif strcmp(mapType, 'LPCD') == 1
401 %               error('LPCD: Interpolation unsuccesful. ...
        Boundaries checked, solution should exist. check code')
402 %
403 %               elseif strcmp(mapType, 'IPC') == 1
404 %               error('IPC: Interpolation unsuccesful. ...
        Boundaries checked, solution should exist. check code')
405 %
406 %               elseif strcmp(mapType, 'HPC') == 1
407 %               error('HPC: Interpolation unsuccesful. ...
        Boundaries checked, solution should exist. check code')
408 %
409 %               end
410 
411         end
412     end
```

```
413     end
414
415 else
416     massFlowValu    = interp2(speeVect, betaVect, ...
            massFlowTabl', N, BETA);
417     effiValu        = interp2(speeVect, betaVect, ...
            effiTabl', N, BETA);
418 end
419
420
421
422 % Step 7: Check for surging
423 %=============================================================
424     % Check for surging
425     if (massFlowValu ≤ surgPoin(1,1)) || (PR ≥ surgPoin(2,1))
426
427         if strcmp(mapType, 'LPCC') == 1
428         warning('LPCC: Map Interpolation successful but ...
                Compressor Surging!')
429
430         elseif strcmp(mapType, 'LPCD') == 1
431         warning('LPCD: Map Interpolation successful but ...
                Compressor Surging!')
432
433         elseif strcmp(mapType, 'IPC') == 1
434         warning('IPC: Map Interpolation successful but ...
                Compressor Surging!')
435
436         elseif strcmp(mapType, 'HPC') == 1
437         warning('HPC: Map Interpolation successful but ...
                Compressor Surging!')
438
439         end
440
441     end
442
443 end
444
445 end
446 % Code used to create plots for checking solutions
447 %=============================================================
448 % plot(WCsline, PRsline, '-x', WCi, PRi, 'o', massFlowValu, ...
        PR,'o'), grid
449 % title('Interpolation of Conditions With Multiple Solutions')
450 % xlabel('WC [kg/s]')
451 % ylabel('PR')
452 % legend('Interpolated Speed Line', 'Initial BETA ...
        Position', 'Interpolated Solution')
453 return
```

D.1.5 Validation

This subsection serves to demonstrate the validity of the methodology and procedures developed for handling the compressor maps. The objective is to demonstrate that the map scaling, correction factors and interpolation procedures produce results that are consistent with those of GSP, under the same conditions. To this end, validation tests were performed at both static sea-level and cruise conditions, for both steady-state and transient operation.

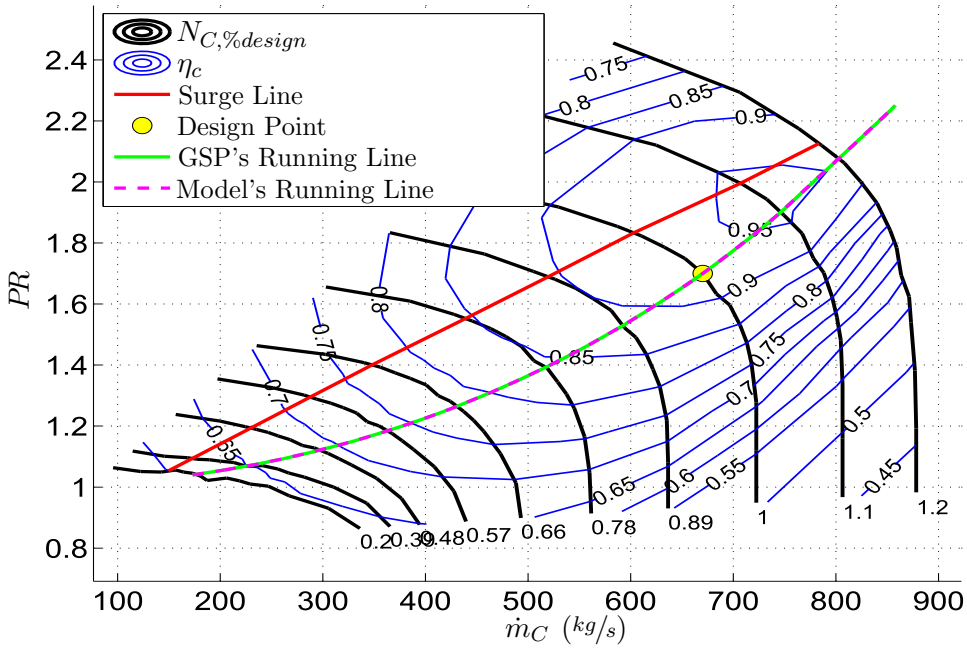
Steady-State, Static Sea-Level Conditions

As an initial validation test, each compressor's running line (corresponding to a steady-state sweep of the engine at static sea-level conditions) was considered, and is depicted in Figures D.15a, D.16a, D.17a and D.18a.

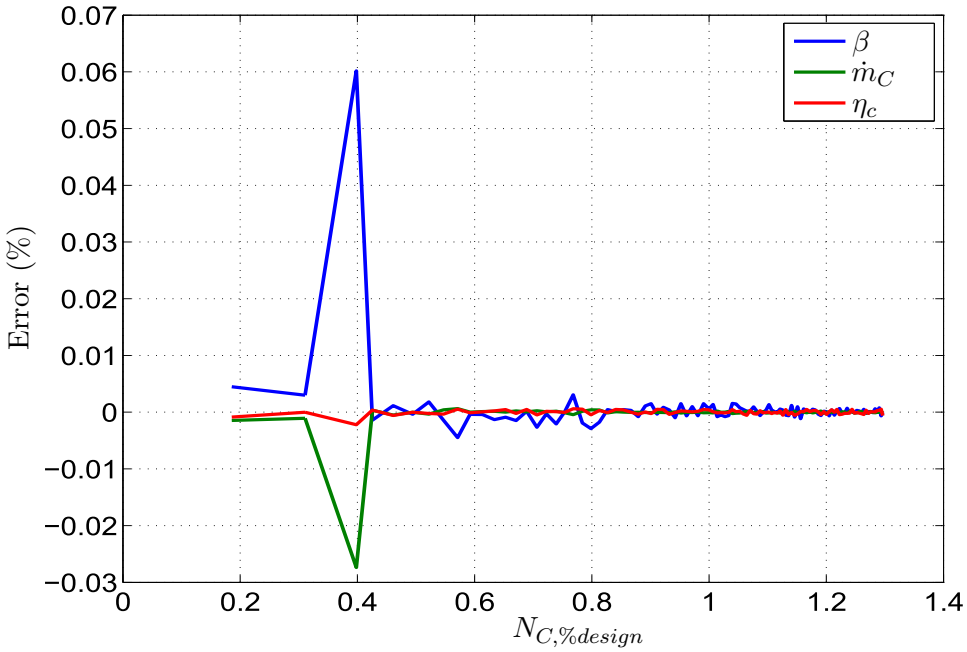
The input values, required for map scaling and interpolation, were obtained from GSP at each point along the running line for each map. These values were imputed into the the procedure developed in this work, who's results were compared to those of GSP. The error between the two models is indicated in Figures D.15b, D.16b, D.17b and D.18b.

The first observation is that, although the errors for all the maps are generally small, the average error increases from the low to high pressure compressor maps. This trend is attributed to discrepancies in map scaling. Remembering that map scaling is performed according to corrected design point parameters, it is crucial that these corrected parameters be calculated accurately. In GSP, the design point parameters are specified as non-corrected terms. These parameters are corrected during a design point calculation of the entire engine. This validation made use of the corrected design point terms, outputted by GSP, to define its scaling factors. GSP's outputs are subject to rounding errors, which introduces inaccuracies into the validation model. The reason why the low pressure compressors are not effected, is because its correction factors, Equation D.3 and D.4, are of unity. This is because at static sea-level conditions, the temperature and pressure entering the low pressure compressors is the same as standard atmospheric conditions. Therefore, the corrected parameters, on the right hand side of Equations D.1 and D.2, at design conditions are the same as the non-corrected design values. These are explicitly known, and hence the interpolation procedure produces accurate results.

However, the intermediate and high pressure compressors, have inlet conditions that differ from the standard pressure and temperature. Thus, the corrected design parameters are required to scale the maps appropriately. But, as already mentioned, these are determined from a design point calculation of the entire engine. These calculated values are rounded off by GSP, and it was necessary to use these values in the validation. Hence, there

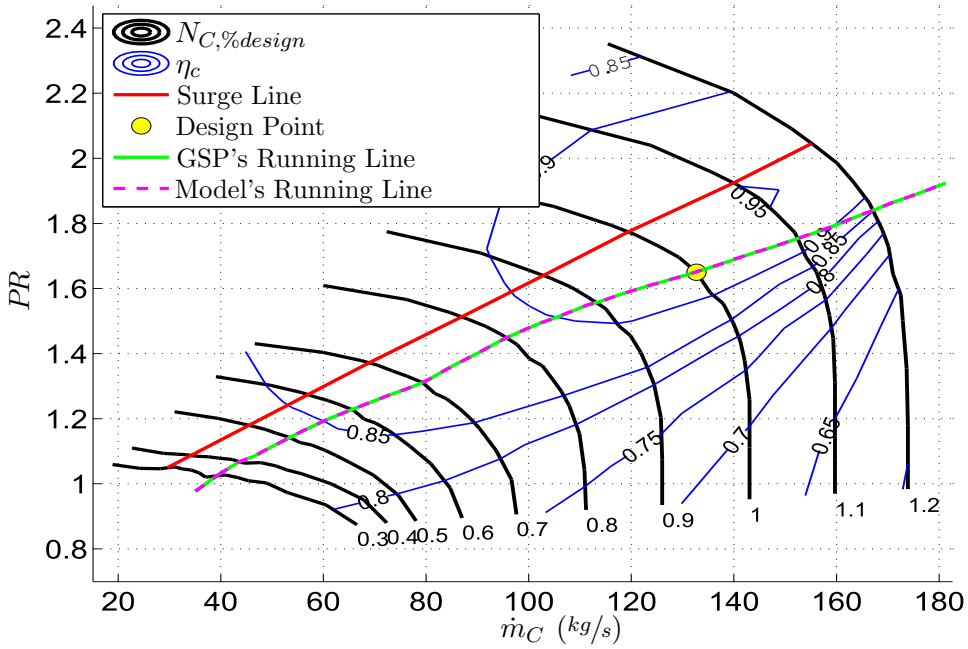


(a) Compressor map with running line

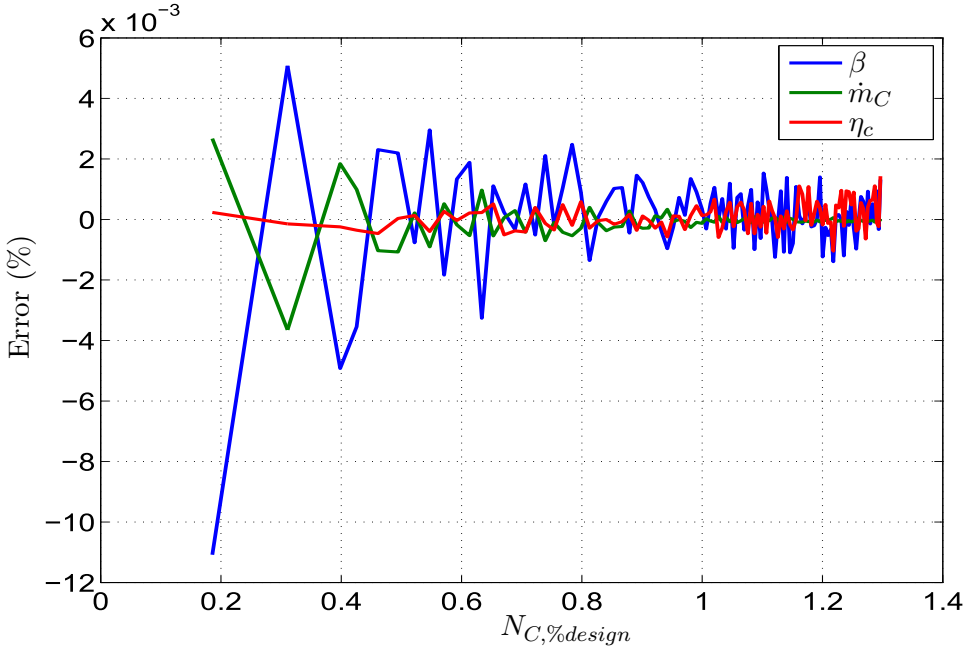


(b) Corresponding interpolation error

Figure D.15: Low pressure compressor map (bypass) Interpolation: Steady-state running line at static sea-level conditions

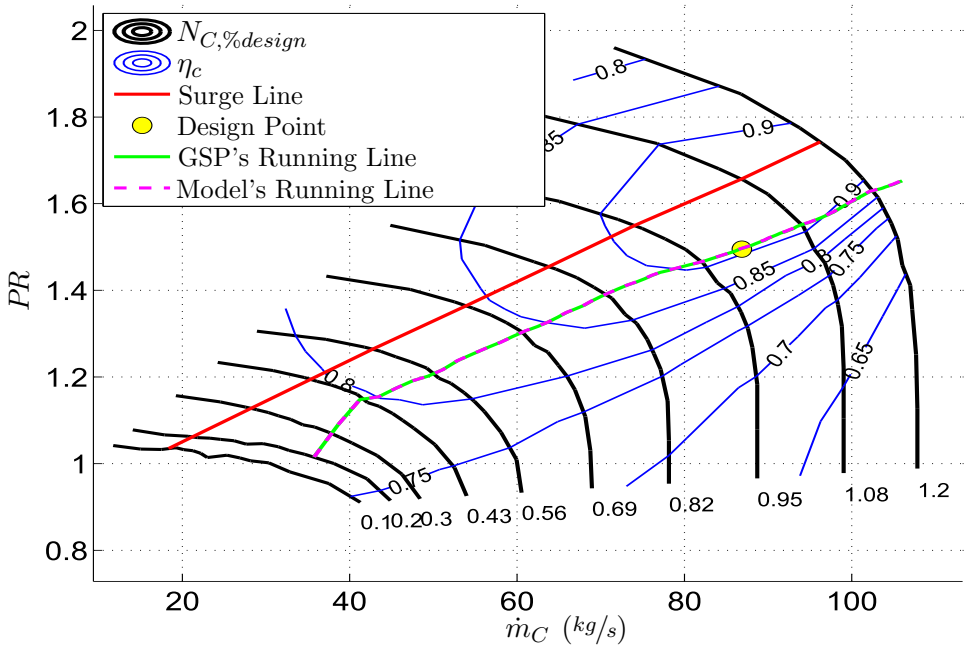


(a) Compressor map with running line

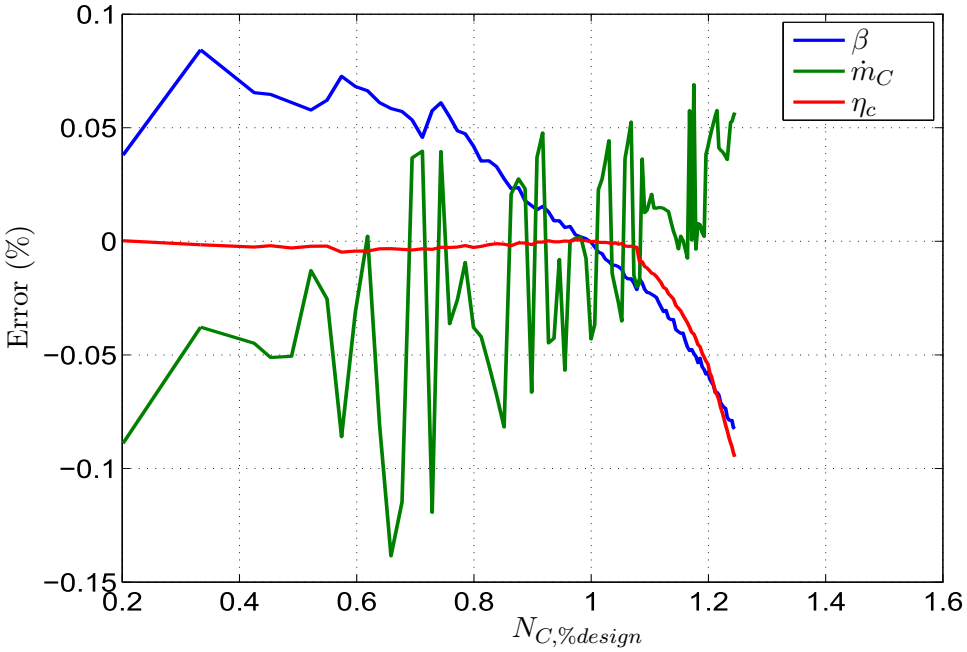


(b) Corresponding interpolation error

Figure D.16: Low pressure compressor map (core) Interpolation: Steady-state running line at static sea-level conditions

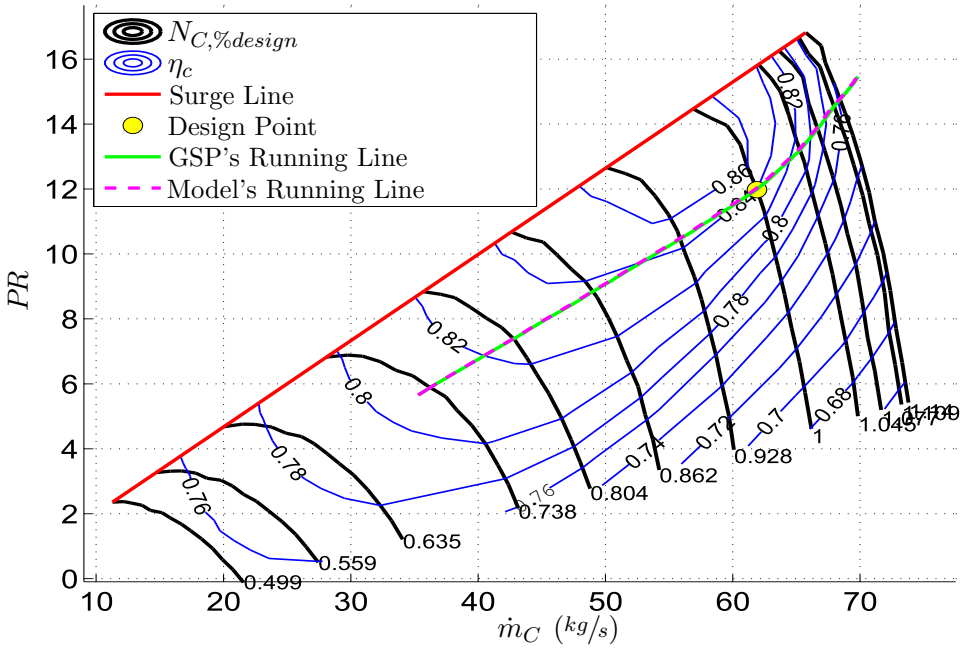


(a) Compressor map with running line

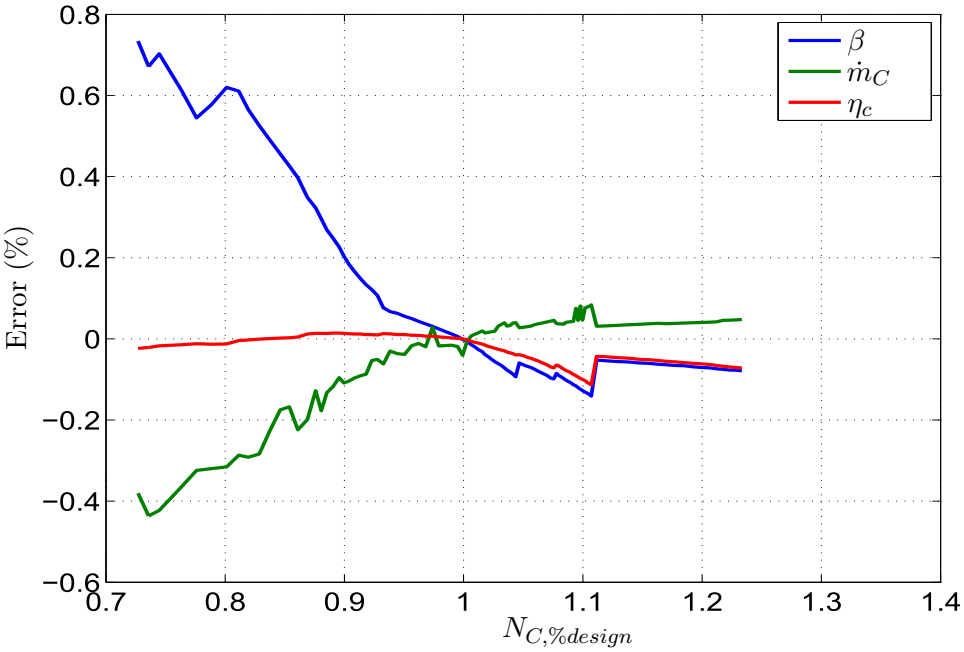


(b) Corresponding interpolation error

Figure D.17: Intermediate pressure compressor map Interpolation: Steady-state running line at static sea-level conditions



(a) Compressor map with running line



(b) Corresponding interpolation error

Figure D.18: High pressure compressor map Interpolation: Steady-state running line at static sea-level conditions

is a slight discrepancy in the scaling of the intermediate and high pressure maps, causing a reduction in accuracy of the interpolated results. Another observation that supports this explanation, is that the error is smallest as it approaches the design speed of the compressor (this is particularly evident in Figure D.18b). As the running line moves away from the design point, in either direction, the discrepancies in map scaling are compounded and hence the interpolation error is amplified. Unfortunately, the only way to mitigate this error is to obtain the exact scaling factors used by GSP, which is not possible.

One of the visually most notable errors is that occurring at $N_{C,\%design} = 0.4$ depicted on Figure D.15b. The cause of this error is not obvious, as Figure D.15a suggests that the running line passes through speed line where no strange anomalies occur. This error is significantly small and therefore an investigation into its possible cause has been neglected.

Finally, linear extrapolation was used to approximate operating points which fall outside the maps bounds. It was expected that extrapolation will introduce large errors, however it appears as though the extrapolation produced sufficiently accurate results. In Figure D.16 there appears to be an increase in error for extrapolations at low speeds, but these are also neglected as they are sufficiently small.

As it stands, it does not seem necessary to remedy the errors produced during the interpolation, as they are sufficiently small and are attributed to rounding errors. In addition, the scaling method used in this work has been popular and well documented in other works (of particular significance is Reference [61]) and improvements would require the knowledge of precisely the scaling method implemented in GSP. However, further validations must be provided to ensure that the error does not increase too drastically when considering transients at cruise conditions.

Transient Cruise Conditions

This subsections serves to demonstrate the adequacy of the compressor map handling procedures developed, at transient cruise conditions. A rather extreme approach was adopted, in that the transient used for validation, resulted in compressor trajectories that ran far away from the compressors' steady-state line, as well as exceeding surge, speed and pressure ratio limits. Such a transient was achieved by adjusting fuel flow schedules, through trial and error, to GSP's sample model. The purpose of validating against such extreme trajectories is to illuminate any vulnerabilities in the procedures developed. The trajectories and their corresponding interpolation errors are depicted in Figures D.19, D.20, D.21 and D.22.

The low pressure map (bypass), Figure D.19, indicates that during this transient the compressor operates close to its steady-state running line, and therefore the interpolation procedure closely matches that of GSP, approxi-

mately within 0.1% . This accuracy is maintained even during extrapolation when the spool speed exceeds that of the maximum speed bounding the map.

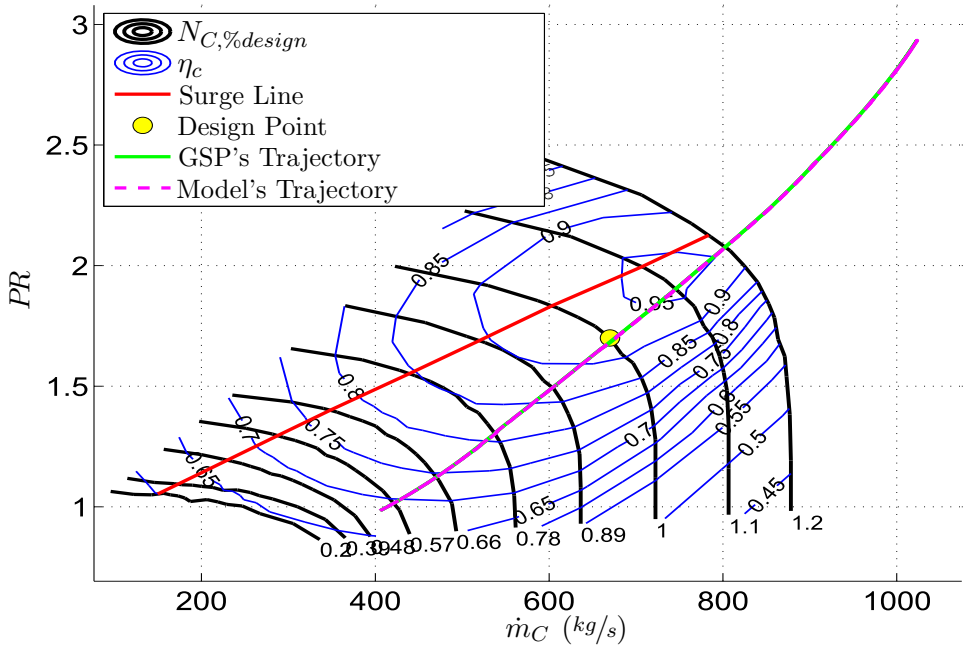
The fan's core map in Figure D.20a, on the other hand, is subjected to a trajectory that surpasses surge, pressure ratio and speed limits. Nonetheless, the resulting discrepancies (Figure D.20b), when compared to GSP, are still very small (well below 1%) even when extrapolation was necessary. The portion of the trajectory exceeding the maximum pressure ratios, yields larger errors. This is attributed to the use of extrapolation. However, as before, extrapolation into high speed region appears to yield fairly accurate results. Overall, the accuracy achieved is satisfactory for the purposes of this work. Similarly, the interpolated parameters of the intermediate pressure compressor, Figure D.21, is also of sufficient accuracy.

The high pressure compressor, Figure D.22, exhibits some interesting results. For the most part, the interpolated results are reasonably accurate. But is significantly large at regions of very low and very high speeds. The errors at high speeds, occur outside the bounds of the map and are attributed to errors resulting from the extrapolation procedure used. An extrapolation procedure was not specifically designed for this purpose, and therefore these errors are not of major concern as the user is not expected to operate the engine in these conditions. However, there is a massive error at $N_{C,\%designe} \approx 0.6$ and at $N_{C,\%designe} \approx 0.7$. Both these areas are contained within the map's bounds, and are therefore of concern.

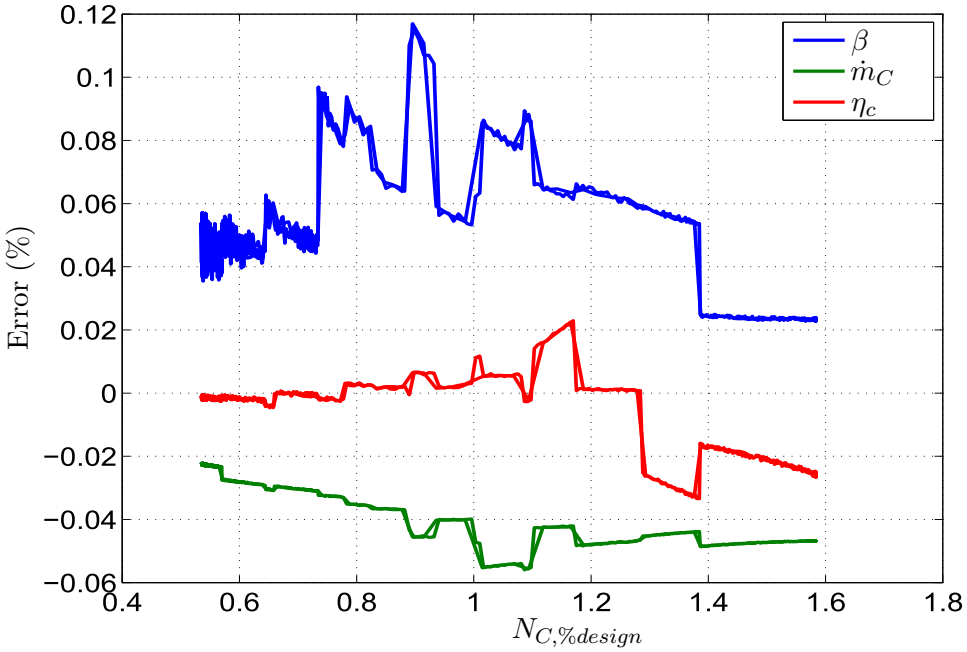
Figure D.23 takes a closer look at this region of concern. From the error clearly visible in the range $\dot{m}_C = (18, 20)$, it is appears as though the interpolation procedure has failed to deal with multiple solutions in the appropriate manner, as done so by GSP. The cause of this error appears to be the same as that already depicted in Figure D.8a, and a similar error is assumed to have occurred at the beginning of the region of error found in the interval $\dot{m}_C = (25, 33)$.

To confirm this hypothesis, it seems obvious that a better estimation of β is required, and the best estimation is to use the β value converged upon by GSP. However, this yielded the exact same result as shown in Figure D.23. In addition, this exact error (along with the addition of some others) still existed, when forcing the interpolation to approach from either end of the speed line. This is shown in Figures D.24a and D.24b. Coincidently, these two figures appear to be identical, and furthermore, the same result occurred when using only MATLAB's built-in interpolation procedure. This does not explain the cause of the error, but nicely demonstrates the effectiveness of the procedure developed in this work, as it conforms much more closely to the trajectory of GSP, when comparing Figure D.23 against Figures D.24a and D.24b. As already mentioned, it is unknown as to what exactly caused this error, and it is also unknown as to how the GSP software deals with such an error to achieve the smooth trajectory it did.

Therefore, if the focus is taken away from these problem areas, a zoomed

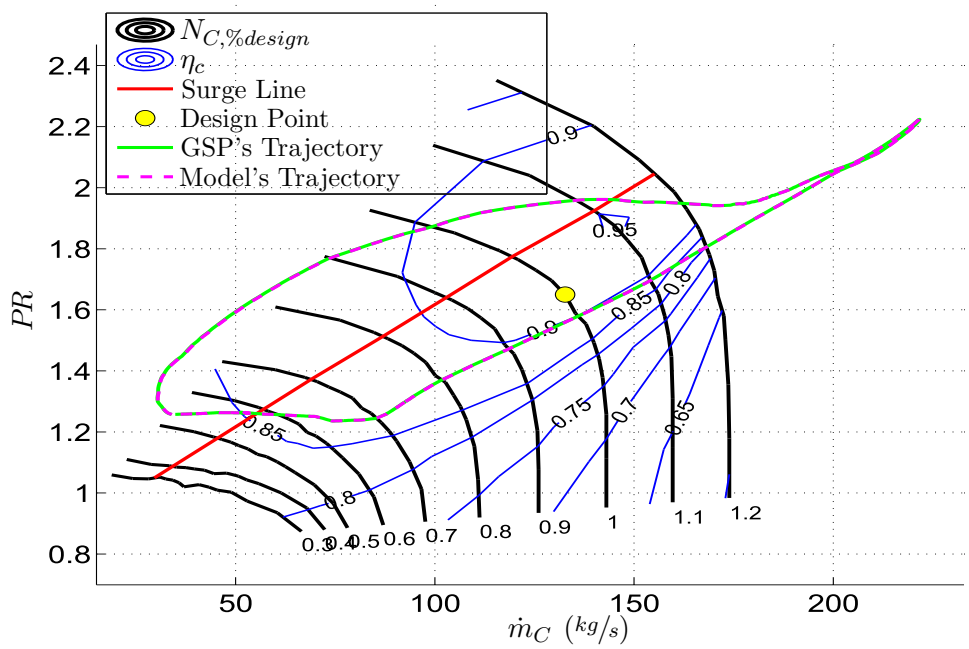


(a) Compressor map with transient trajectories

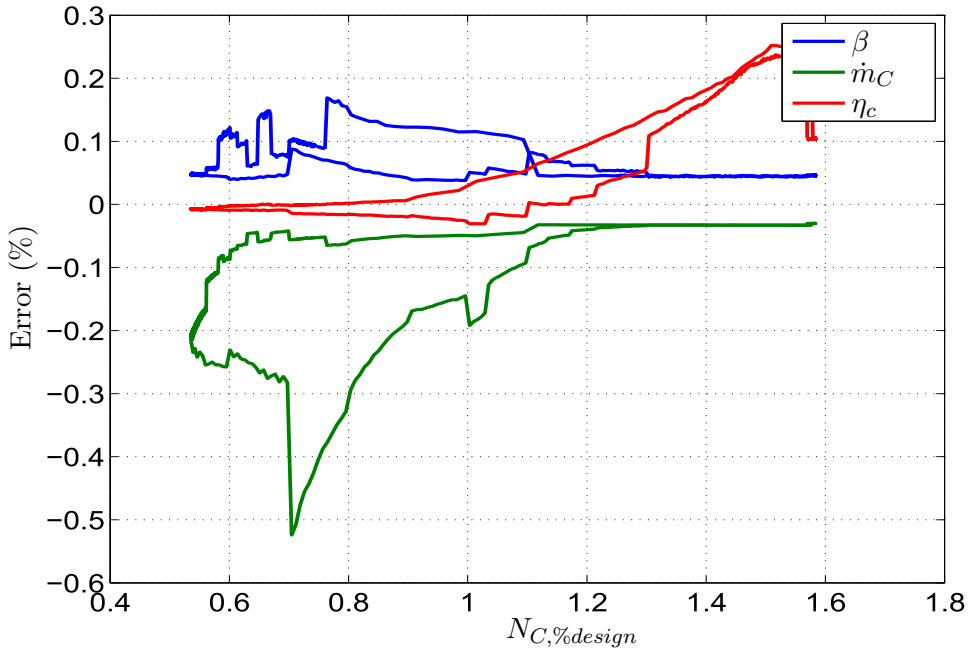


(b) Corresponding interpolation error

Figure D.19: Low pressure compressor map (bypass) Interpolation: Transient trajectories at cruise

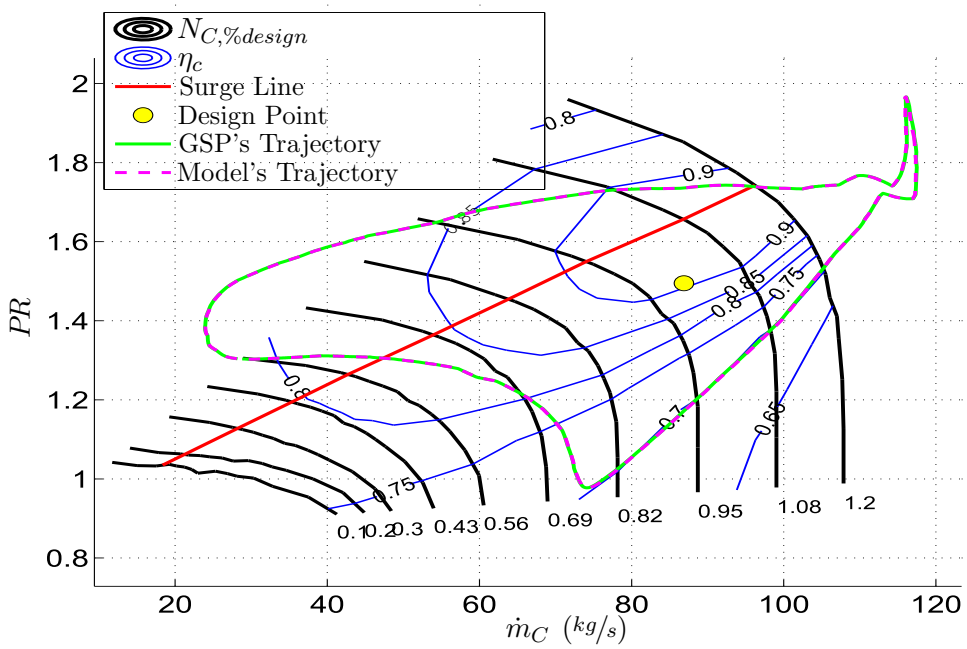


(a) Compressor map with transient trajectories

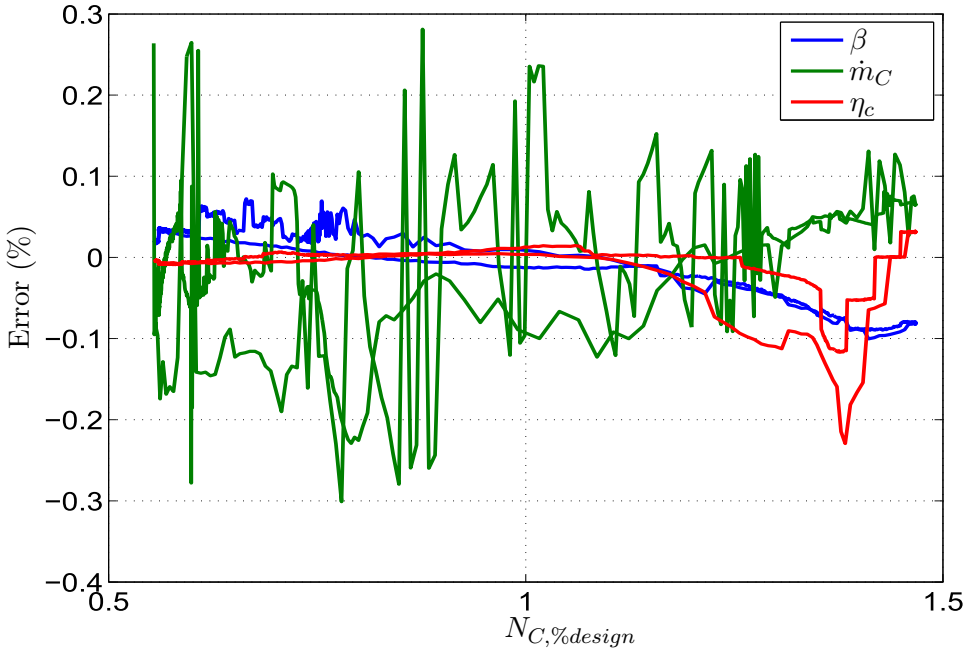


(b) Corresponding interpolation error

Figure D.20: Low pressure compressor map (core) Interpolation: Transient trajectories at cruise

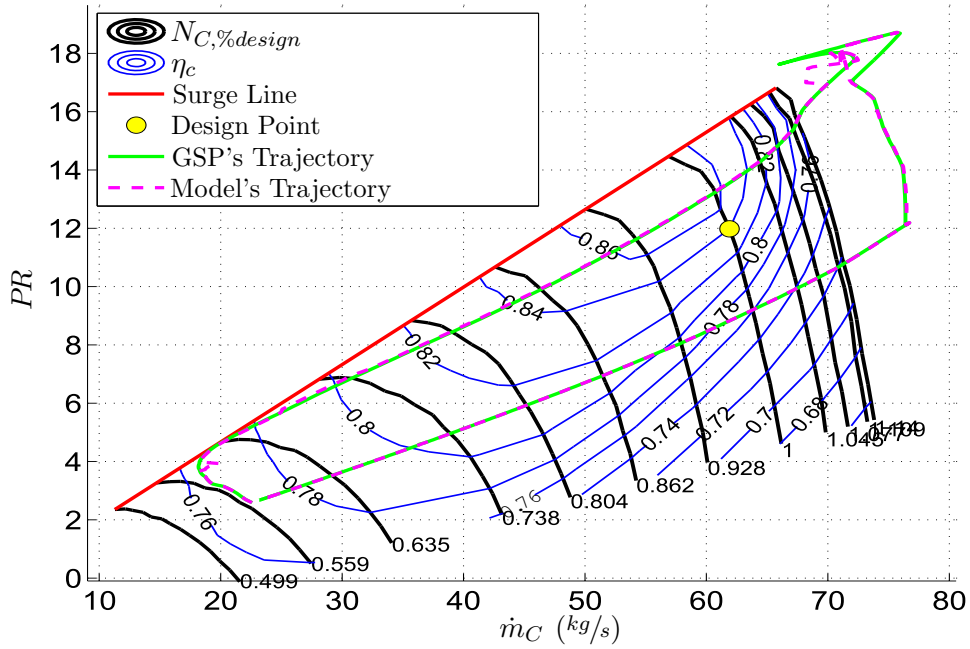


(a) Compressor map with transient trajectories

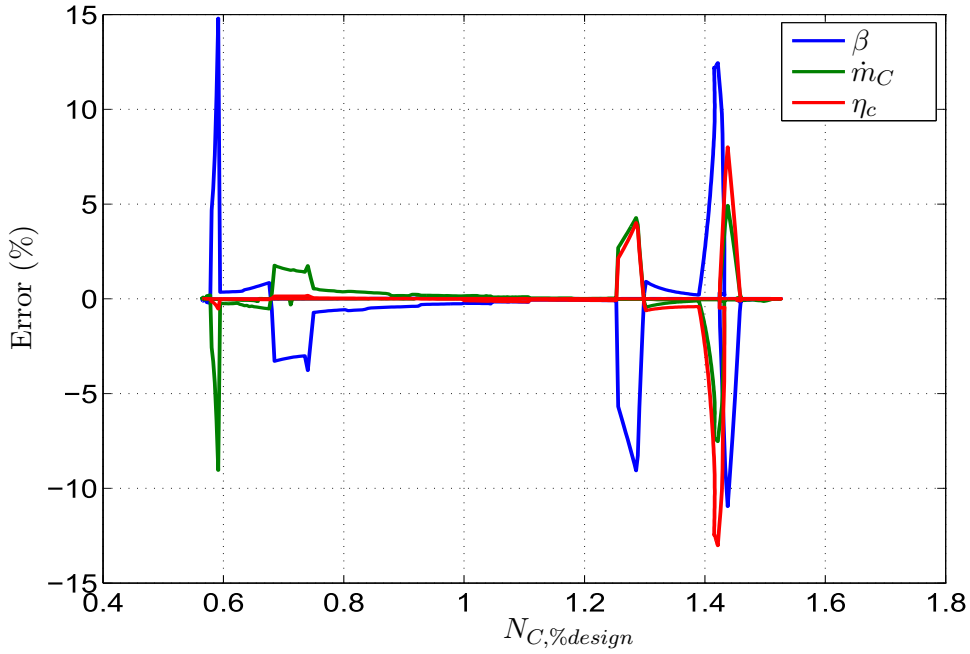


(b) Corresponding interpolation error

Figure D.21: Intermediate pressure compressor map Interpolation: Transient trajectories at cruise



(a) Compressor map with transient trajectories



(b) Corresponding interpolation error

Figure D.22: High pressure compressor map Interpolation: Transient trajectories at cruise

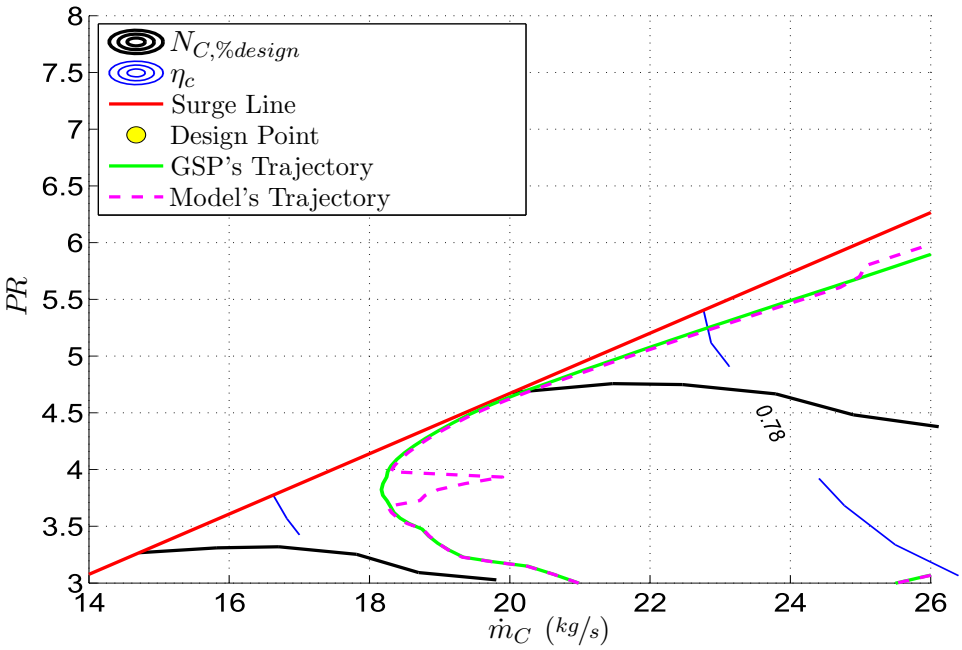


Figure D.23: High pressure compressor low speed interpolation error: Zoomed in on Figure D.22a

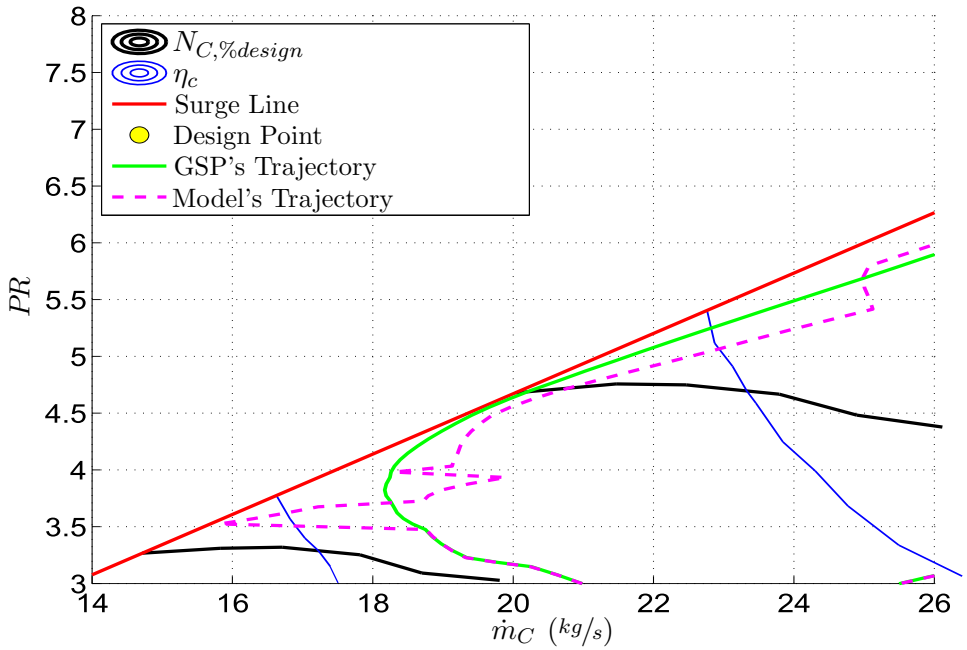
in view of the interpolation of the high pressure map yields sufficiently accurate results, as depicted in Figure D.25

Therefore, if the error scenario of Figure D.23 is assumed to occur only extremely rarely (as has been the experience thus far in this work) and the compressor is assumed to operate within its map's bounds, then the map handling procedures developed in this work provide sufficient accuracy and are valid.

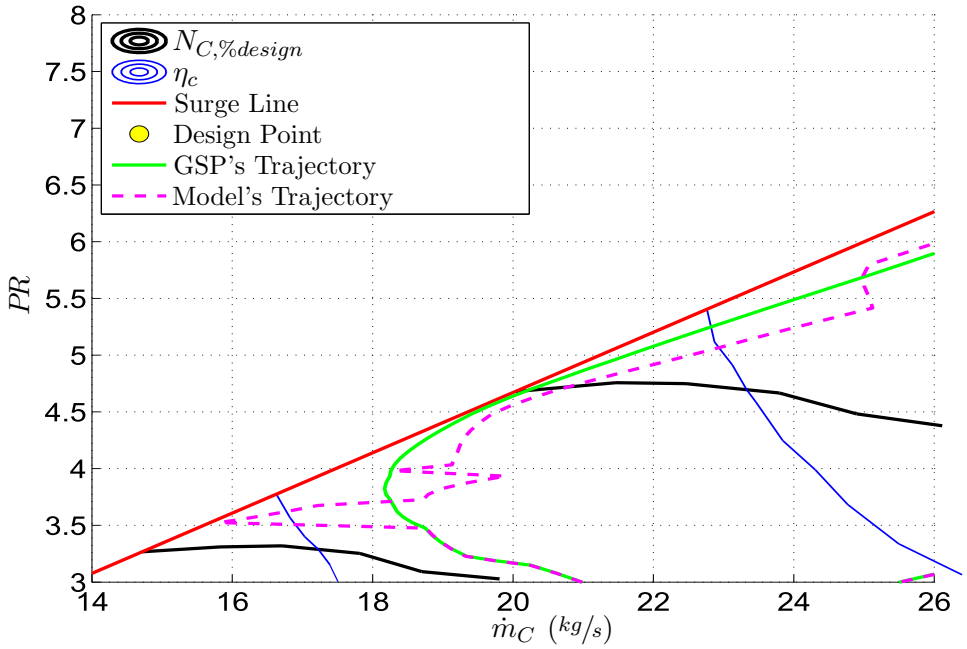
D.2 Turbine Performance Maps

Like with compressors, maps are used to describe turbine performance in terms of the parameters given by Equations D.1 to D.5. As with compressors, information on turbine performance is kept proprietary by engine manufacturers and therefore it is necessary to modify existing maps for use in engine simulations. There are many similarities regarding performance map handling between compressor and turbines, therefore this discussion regarding turbine map handling is limited to avoid repetition.

Typically, as with the compressor, maps are used to depict the relationship between speed, pressure ratio, mass flow rate and efficiency, which are corrected according to inlet conditions. Occasionally, pressure ratio is substituted by specific work, however the former parameter is used here. The turbine parameter relationships may be depicted using twin maps, $\eta_t = f\langle PR \rangle$



(a) High pressure compressor low speed interpolation error: Forced interpolation from $\beta = 0$



(b) High pressure compressor low speed interpolation error: Forced interpolation from $\beta = 1$

Figure D.24: High pressure compressor low speed interpolation error: Forced interpolation direction

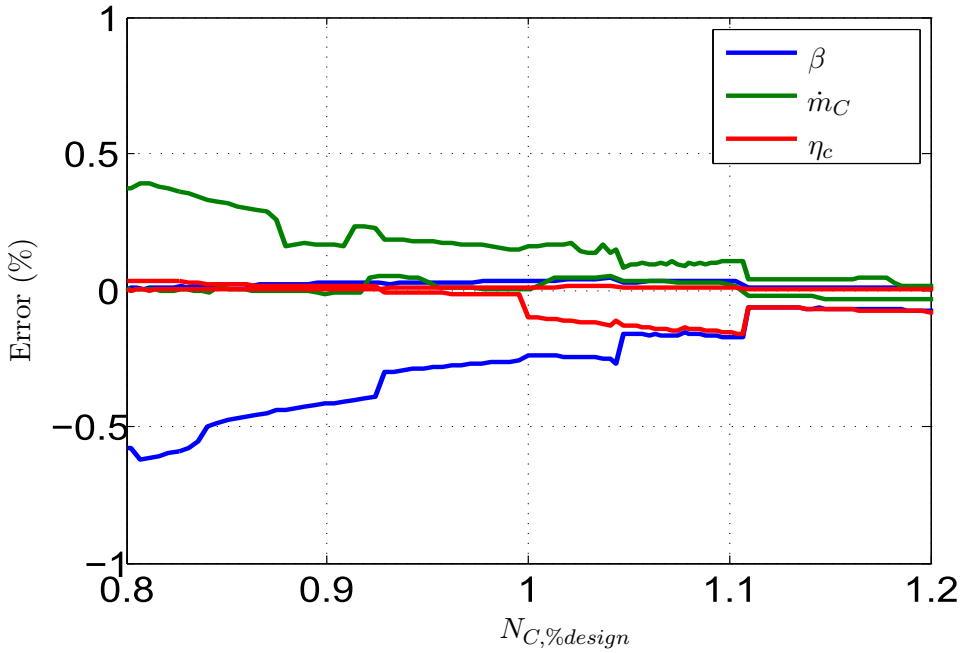


Figure D.25: High pressure compressor low speed interpolation error: Zoom of accurately interpolated region of Figure D.22b

and $\dot{m}_C = f\langle PR \rangle$, or a single map with speed and efficiency contours. In the latter format, pressure ratio and efficiency are related to $(\dot{m}_C \times N_C)$ for graphical purposes. This is because the map's speed and efficiency isolines become very congested without the use of the N_C multiplier, resulting in maps that are difficult to read and interpret. The single map representation has been selected for this work as it is consistent with GSP and the compressor map representations (which allows for easy reuse and development of code).

Figures D.26 and D.27 depict the generalised maps used for modelling the low and high pressure turbines respectively. Like with the compressor maps, these maps were constructed from data obtained from GSP's BIG-FAN sample model. The vertical axis represents the inverse of the pressure ratio, this simply makes the map more readable. Unlike the compressor, the turbine does not operate in an adverse pressure gradient; thus there is no surge line. However, each speed isoline asymptotes at a certain $(\dot{m}_C \times N_C)$ value. At this value the turbine chokes, preventing an increase in mass flow despite any increase in pressure ratio. Choking occurs when the flow reaches sonic conditions within the turbine. Subsequently, at choke, an increase in pressure ratio will result in a decrease in operational efficiency of the turbine. Sometimes by removing the N_C multiplier on the horizontal axis, these asymptotes converge on a single mass flow rate. This then referred to as the common choke line of the turbine, and dictates the maximum flow rate

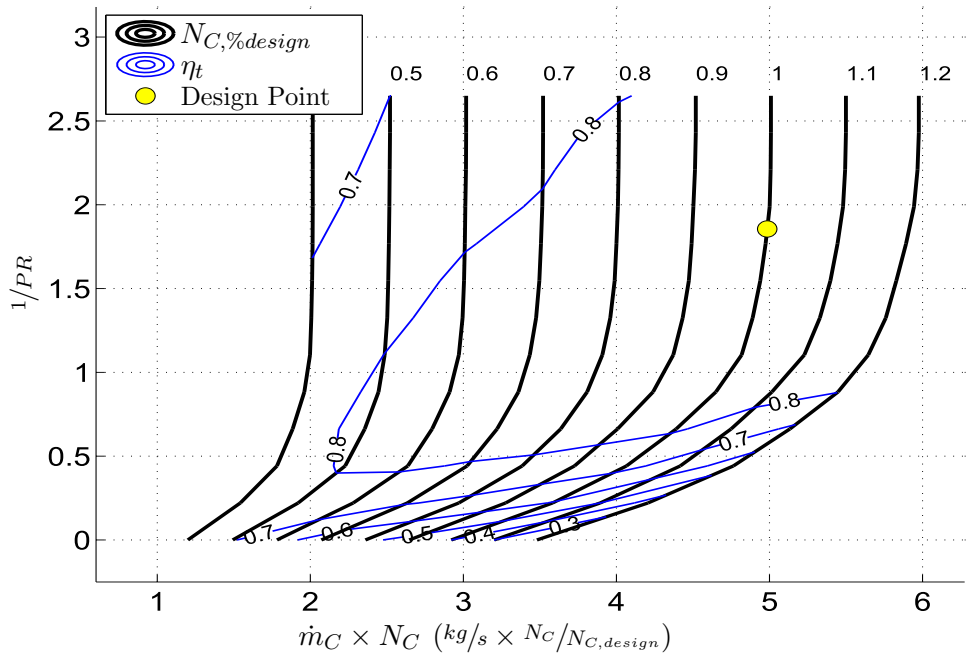


Figure D.26: Low pressure turbine map

through the turbine for all operational speeds.

Like with the compressor, these representations have to be translated into a computational friendly format. This is discussed in the following subsections.

D.2.1 Turbine Map Handling

As with compressor maps, turbine characteristics may be represented by curve-fits or discretised into look-up tables, for computational use. Again, it is very difficult to find curve-fits that conform to the shapes of the speed isolines of turbine maps. Approximate methods have been documented by Moore and Greitzer [120] and Flack [33], but suffer similar issues as those used for estimating compressor speed isolines. Piecewise functions are an alternative that may be used to represent the turbine characteristics accurately, but this becomes a very time consuming task when preparing turbine maps for computational use.

Instead, like with the compressor maps, smoothed turbine maps may be discretised and converted into look-up tables for inter/extrapolation. A popular method for discretization follows the work of Kurzke [82] and has been implemented in the commercial software package Smooth T [127].

In the case of turbine performance maps, unlike compressors, there are distinct values of mass flow rate corresponding to pressure ratio (i.e. mass flow rate may be represented as a function of pressure ratio $\dot{m}_C = f\langle PR \rangle$).

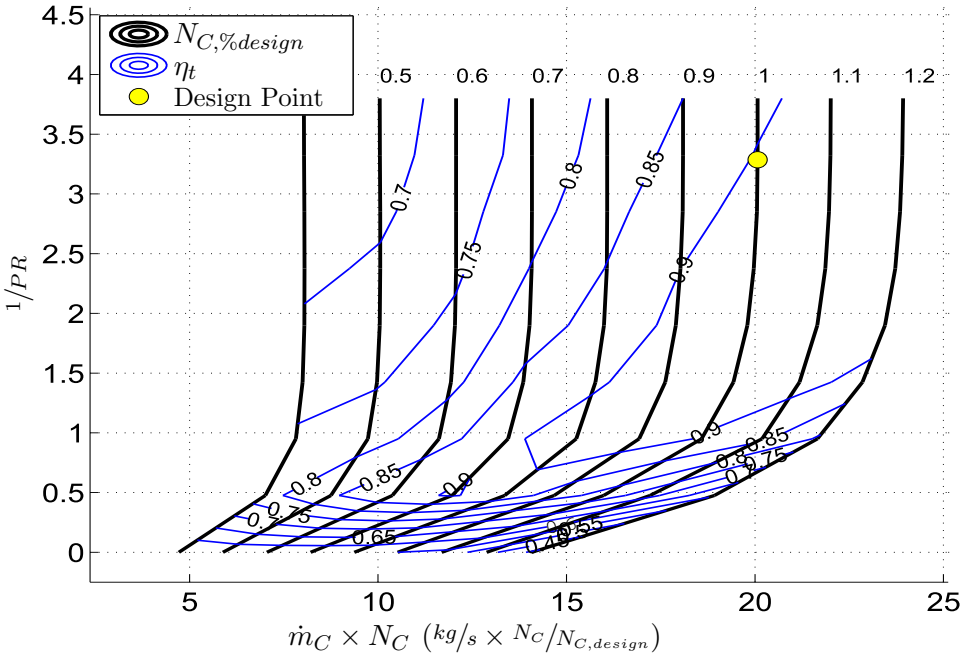


Figure D.27: High pressure turbine map

Therefore, turbine maps are not subject to the same interpolation issues as those encountered by compressor maps. However, Kurzke still advises the use of an auxiliary coordinate system, β grid, to aid in the optimization of the number of points to be discretised and to minimize interpolation errors.

Kurzke observed that, as is evident in Figures D.26 and D.27, that efficiency isolines become more closely packed together at low corrected speeds and pressure ratios. Therefore, these regions of the map require a denser discretization to yield accurate results when interpolating. However, a far fewer number of points are needed to achieve the same accuracy when interpolating at higher speeds and pressures. On this basis, Kurzke suggested the use of an auxiliary coordinate system, which would naturally aid the discretisation process, to best accommodate the turbine characteristics.

To do so, β isolines are superimposed over a turbine map representing pressure ratio as a function of corrected speed, $PR = f\langle N_C \rangle$. Two polynomials curve-fits are then used to enclose the map data, as illustrated in Figure D.28 [1]. The first, $\beta = 0$, connecting the points of maximum pressure ratio (of each speed isoline), and the second, $\beta = 0$, connecting points of minimum pressure ratio. The remaining beta lines are inserted between these boundaries, as illustrated in Figure D.29. They are equispaced automatically by evenly incrementing the values of the polynomial coefficients. As can be seen in Figure D.30, the β lines are more concentrated where the

[1] It is important to note here that the maps used in this explanation are fictitious, and simply serve to show the general shape of a real turbine map

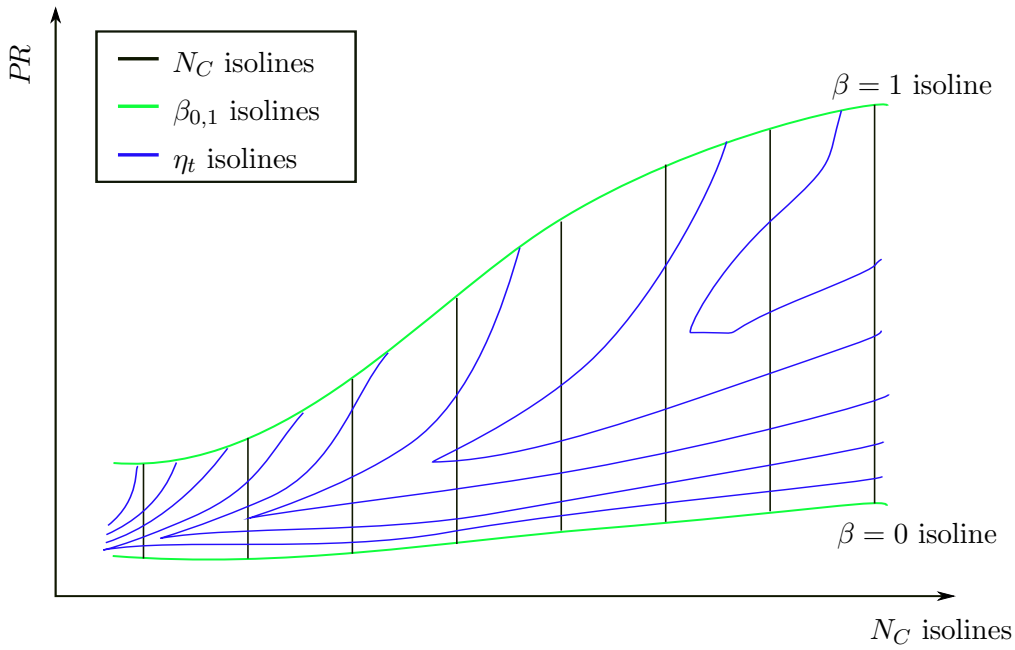


Figure D.28: Turbine map β coordinate system generation: β boundaries

efficiency contours are closely packed together. Look-up tables may then be constructed from the intersections of the polynomial β curves and the speed isolines. Therefore, there are a larger number of data points representing the "congested" portions of the map, improving interpolation accuracy.

However, the turbine map data obtained from GSP's BIGFAN sample model, gives minimum and maximum pressure ratios that remain constant across all speeds of the engine. This clearly evident from Figures D.26 and D.27. In this scenario the β curves become horizontal lines across the compressor map, representing pressure isolines, as depicted in Figure D.31a.

In this case, the look-up tables generated may be optimised, to reduce the number of data points and improve interpolating accuracy, by bunching up the number of β lines used at low pressure ratios. This aids better representation of the map by increasing the density of data points where efficiency gradients are largest, as depicted in Figure D.31b.

However, when obtaining the turbine map data from GSP, it was evident that the β lines were equispaced, as in Figure D.31c. Therefore the β curves served no real purpose in optimising the look-up tables generated for interpolation. Nonetheless, equispaced β curves were also used in this work, because it is GSP's model against which this model is validated. Subsequently, the β and speed isoline intercepts are used to generate look-up tables of the form given in Table D.1, which is then used to determine turbine performance characteristics given pressure ratio and speed.

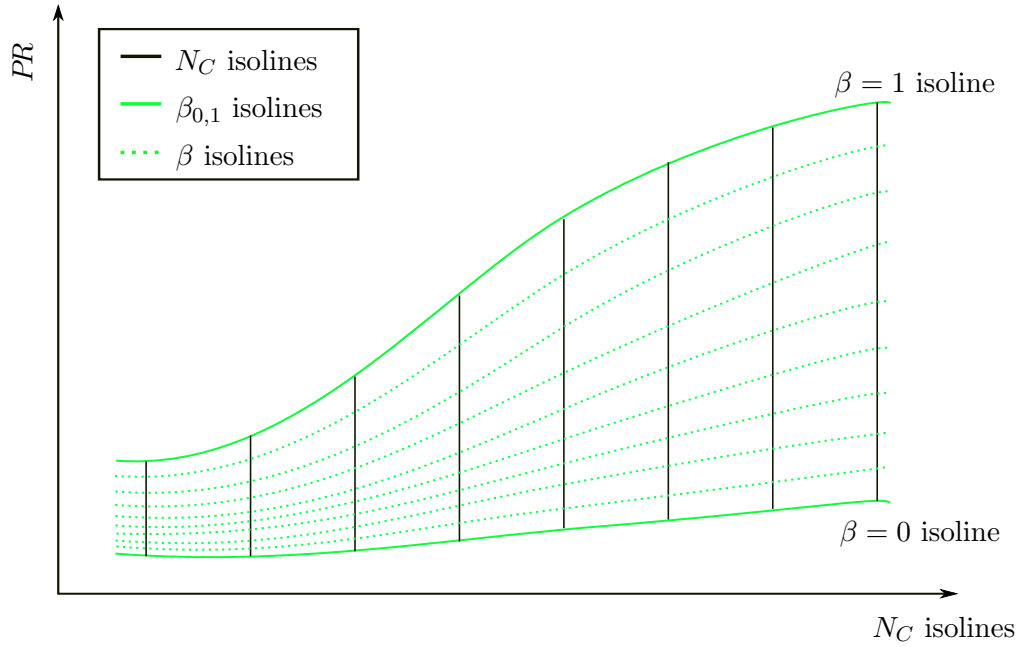


Figure D.29: Turbine map β coordinate system generation: β lines

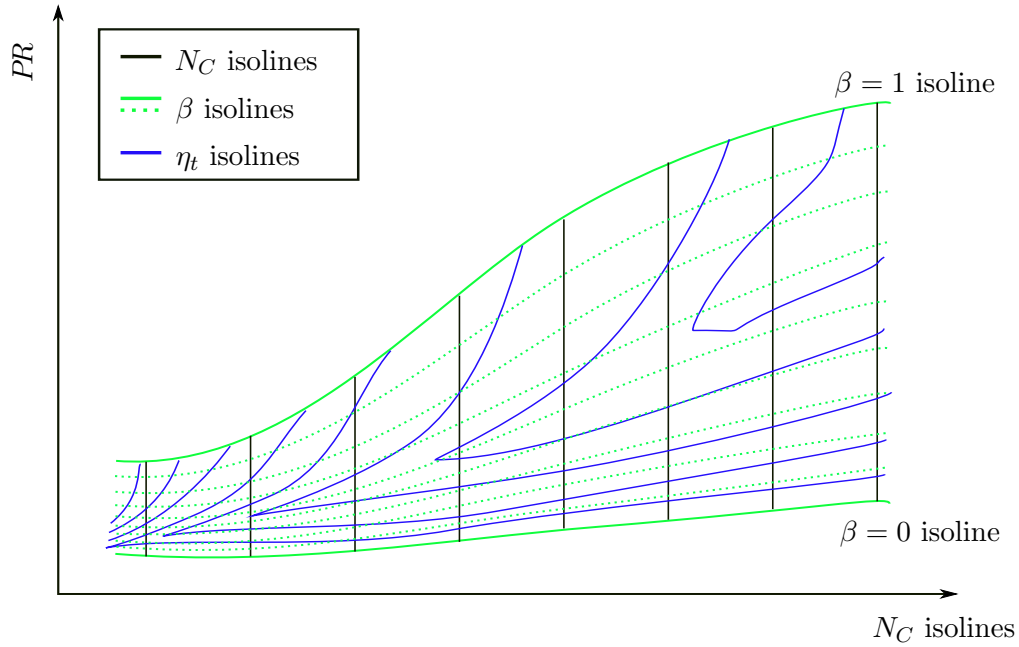
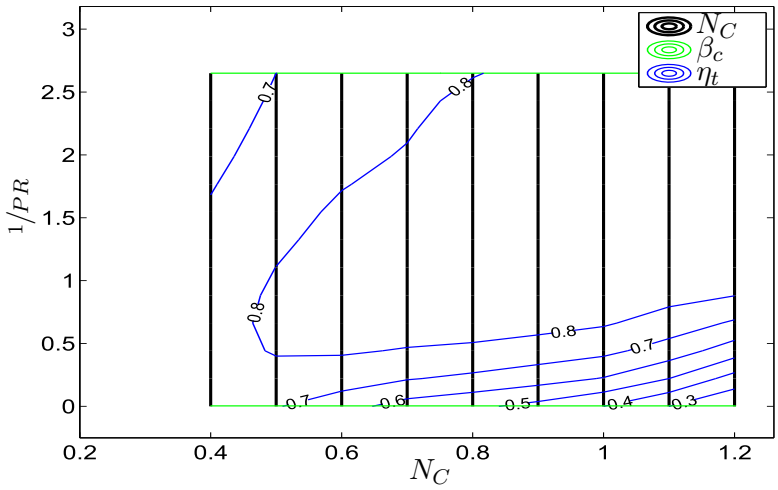
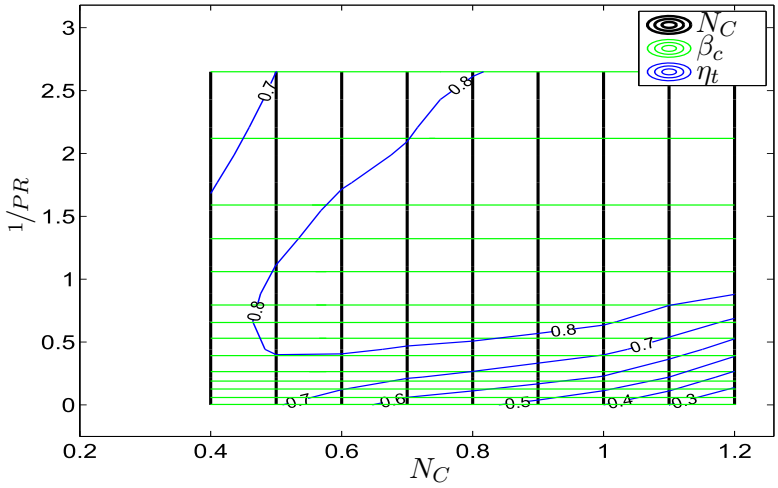


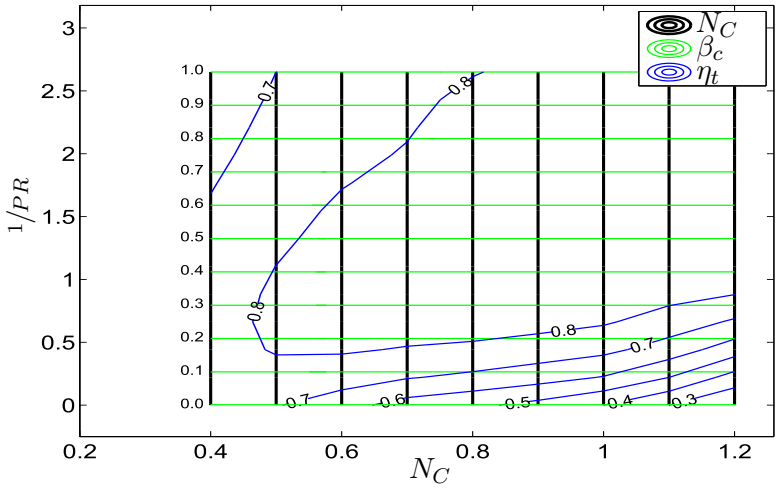
Figure D.30: Turbine map β coordinate system generation: β lines superimposed over η_t contour



(a) Minimum and maximum pressure, constant across all speeds, results in horizontal β lines representing pressure isolines



(b) Strategically placed β isolines superimposed over turbine map for optimisation of data points and interpolation accuracy



(c) Equispaced β isolines superimposed over turbine map

Figure D.31: β generation of turbine map look-up tables from GSP data

D.2.2 Interpolation of Turbine Maps

When performing transient engine simulations, it is common that either turbine β or pressure ratio will be used for inclusion in a state vector. In this work, pressure ratio is used and therefore it becomes necessary to define the turbine's performance at a given operating speed and pressure ratio. Unlike the compressor, it is very easy to interpolate turbine maps, because mass flow rate has a distinct value for a given pressure ratio along a speed line. Therefore, a simple 2D linear interpolation function (provided by MATLAB) was used to obtain the turbine's mass flow rate and efficiency from the look-up tables generated.

D.2.3 Turbine Map Scaling

The turbine maps were scaled using the exact same method used to scale the compressor maps. Therefore the reader is referred to Section D.1.3.

D.2.4 MATLAB Routines

The turbine maps were stored in a very similar manner to those of the compressor maps. In addition, the interpolation routine is very simple. Therefore code listing are included for the sake of completeness.

Turbine Map Library

Included in this model is a library of two generalised maps, obtained from GSP's BIGFAN sample model, to be used when modelling a typical high bypass turbofan engine. These include a low and high pressure turbine maps, which are depicted in Figures D.26 and D.27, respectively.

These maps were each discretised to obtain tables of the format depicted in Table D.1. These tables are stored in an m-file, as demonstrated in Code Listing D.1.

The actual table entries have been omitted from this code listing to save space in the document. In addition, only the supporting code and comments for one map has been included, as the remaining code is just a repeat of that entry. Therefore, this architecture allows for easy expansion of the compressor map library.

This library is presented in Code Listing D.3, and due to its similarity to the compressor map library, a description of the code listing has been omitted to avoid repetition.

Code Listing D.3: MATLAB Function TurbMaps.m: Library of turbine maps

```
1 %=============================================================
2 %                          Turbine Map Data
```

```
3  %=============================================================
4  %
5  % NAME:             TurbMaps.m
6  %
7  % AUTHOR:           D. Sanders, University of Cape Town
8  %
9  % DATE:             15 October 2012
10 %
11 % FUNCTION:         [betaVect, betaRange, speeVect, ...
       massFlowTabl, presRatiRange, effiTabl, mapDP] = ...
       TurbMaps (mapType)
12 %
13 % PURPOSE:          Provide data representing generalised ...
       turbine maps which may be scaled according to engine ...
       design point values
14 %
15 % INPUTS:
16 %
17 %   mapType         = String use to select appropriate ...
       compressor map for simulation ('LPT' 'IPT' or 'HPT')
18 %
19 % OUTPUTS:
20
21 %   betaVect        = Vector of beta grid values           [-]
22 %   betaRange       = Min and max beta values              [-]
23 %   speeVect        = Vector of speed isolines      [%Ndesign]
24 %   massFlowTabl    = Corrected mass flow table         [kg/s]
25 %   presRatiRange   = Min and max pressure ratio           [-]
26 %   effiTabl        = Efficiency table                [%/100]
27 %
28 % SUBROUTINES:      None
29 %
30 % REFERENCES:
31 %
32 %   Data Obtained from:
33 %       GSP BIGFAN example model. NLR, Gas Turbine Simulation
34 %       Program 2011.
35 %
36 % For coding conventions, please refer to:
37 %
38 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
39 %   ver 1.5. October 2002.
40 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
41 %=============================================================
42
43 function [betaVect, betaRange, speeVect, massFlowTabl, ...
       presRatiRange, effiTabl, mapDP] = TurbMaps (mapType)
44
45 % Select generalised map to be used
46 switch mapType
47
48 %-------------------------------------------------------------
49 %             LPT - Generalised low pressure turbine map
50 %-------------------------------------------------------------
```

```
51
52 case {'LPT'}
53
54 % Beta Refernce Grid Values
55 betaVect = [...];
56
57 betaRange = [betaVect(1) betaVect(end)];
58
59 % Constant Speed Lines
60 speeVect = [...];
61
62 % Mass Flow Look-Up Tables (Rows, Cols) = (Constant Speed, ...
       Constant Beta)
63 massFlowTabl = [... ];
64
65 % Efficiency Look-Up Tables (Rows, Cols) = (Constant Speed, ...
       Constant Beta)
66 effiTabl = [...];
67
68 % Press Ratio Look-Up Tables (min-pres-ratio; max-pres-ratio)
69 presRatiRange = [min max];
70
71 % Specify the design point of the unscaled map
72 % mapDP = [beta Nc Wc ETA PR];
73
74 Nd = ...;
75 BETAd = ...;
76
77 PRd     = interp1(betaRange, presRatiRange, BETAd);
78 WCd     = interp2(speeVect, betaVect, massFlowTabl' , Nd, ...
       BETAd);
79 ETAd    = interp2(speeVect, betaVect, effiTabl'     , Nd, ...
       BETAd);
80
81 mapDP = [BETAd Nd WCd ETAd PRd];
82
83 %================================================================
84 % Repeated for high, intermediate and high pressure maps
```

Turbine Map Interpolation

The interpolation routine used for determining the turbines off-design operating point is fairly simple. The routine checks where extrapolation is necessary and issues warnings to the user, where appropriate, to give information as to how the map bounds may have been exceeded during the interpolation. This is important for diagnosing any problems with the model or its external conditions. The routine and its working are presented in Code Listing D.4. A flow chart of the routine has not been included due to its simplicity.

Code Listing D.4: MATLAB Function TurbMapsInterpPR.m: For interpolating turbine maps given N_C and PR

```
%============================================================
%               PR Turbine Map Interpolation Scheme
%============================================================
%
% NAME:             TurbMapsInterpPR.m
%
% AUTHOR:           D. Sanders, University of Cape Town
%
% DATE:             12 October 2012
%
% FUNCTION:         [BETA massFlowValu effiValu] = ...
    TurbMapsInterpPR (N, PR, BETAi, mapType, DP)
%
% PURPOSE:          Interpolate compressor off-design ...
    operating parameters given the pressure ratio
%
% INPUTS:
%
%   N               = Spool Speed                       [rpm]
%   PR              = Pressure Ratio                     [-]
%   mapType         = The type of turbine map to use [string]
%   DP              = Design point parameters of compressor
%
% OUTPUTS:
%
%   BETA            = Beta grid interpolated value       [-]
%   massFlowValu    = Interpolated corrected mass flow [kg/s]
%   effiValu        = Interpolated  efficiency         %/100]
%
% SUBROUTINES:      TurbMaps.m
%
% REFERENCES:
%
%   Data Obtained from:
%       GSP BIGFAN example model. NLR, Gas Turbine Simulation
%       Program 2011.
%
% For coding conventions, please refer to:
%
%   Johnson, Richard. "MATLAB Programming Style Guidlines".
%   ver 1.5. October 2002.
%   http://www.mathworks.com/matlabcentral/fileexchange/2529
%============================================================
%% Turbine Map Off Design operation Predictor

function    [BETA massFlowValu effiValu] = TurbMapsInterpPR ...
    (N, PR, mapType, DP)

%% Map Preparation

% Fetch appropriate map data
```

```
49 [betaVect, betaRange, speeVect, massFlowTabl, ...
       presRatiRange, effiTabl, mapDP] = TurbMaps (mapType);
50
51 if size(DP) ≠ size(mapDP)
52     DP = DP';
53 end
54
55 % Invert Pressure Ratio to suite maps i.e. Maps use and ...
       inverted PR
56 PR = 1/PR;
57 DP(5) = 1/DP(5);
58
59 % Calculate Map Scaling Factors
60 SF = DP ./ mapDP;
61 SF(5) = (DP(5) - 1) / (mapDP(5) - 1);
62
63 % SF = [SFbeta SFnc SFwc SFeta SFpr]
64 betaVect        = SF(1) * betaVect;
65 speeVect        = SF(2) * speeVect; % Changes %-->rpm
66 massFlowTabl    = SF(3) * massFlowTabl;
67 effiTabl        = SF(4) * effiTabl;
68 presRatiRange    =(SF(5)    * presRatiRange) - SF(5) + 1;
69
70
71 %% Check Map Bounds
72 % Check if Spool Speed Falls within in Map Bounds
73 % (N > max(speeVect))
74 % (N < min(speeVect))
75 % (PR > presRatiRange(2))
76 % (PR < presRatiRange(1))
77
78 if ((N > max(speeVect)) || (N < min(speeVect)) || (PR > ...
       presRatiRange(2)) || (PR < presRatiRange(1)))
79
80 if ((N > max(speeVect)) || (N < min(speeVect)))
81
82     % Extrapolate speedline
83     WCsline = interp1(speeVect, massFlowTabl, N,  'linear', ...
           'extrap');
84     ETAsline = interp1(speeVect, effiTabl, N,  'linear', ...
           'extrap');
85
86     if N > max(speeVect)
87
88     warning_message = strcat(mapType, ': Interpolation out ...
           of turbine map bounds: N too high');
89     warning(warning_message)
90
91     elseif N < min(speeVect)
92
93     warning_message = strcat(mapType, ': Interpolation out ...
           of turbine map bounds: N too low');
94     warning(warning_message)
95
```

```
96      end
97
98  else
99
100     % Interpolate speedline
101     WCsline = interp1(speeVect, massFlowTabl, N,  'linear');
102     ETAsline = interp1(speeVect, effiTabl, N,  'linear');
103
104 end
105
106
107 if ((PR > presRatiRange(2)) || (PR < presRatiRange(1)))
108
109     % Extrapolate look-up tables
110     BETA = interp1(presRatiRange, betaRange, PR, 'linear', ...
            'extrap');
111     massFlowValu = interp1(betaVect, WCsline, BETA, ...
            'linear', 'extrap');
112     effiValu = interp1(betaVect, ETAsline, BETA, 'linear', ...
            'extrap');
113
114     elseif PR > presRatiRange(2)
115
116     warning_message = strcat(mapType, ': Interpolation out ...
            of turbine map bounds: 1/PR too high');
117     warning(warning_message)
118
119     elseif PR < presRatiRange(1)
120
121     warning_message = strcat(mapType, ': Interpolation out ...
            of turbine map bounds: 1/PR too low');
122     warning(warning_message)
123
124 else
125
126     % Interpolate look-up tables
127     BETA = interp1(presRatiRange, betaRange, PR, 'linear');
128     massFlowValu = interp1(betaVect, WCsline, BETA, 'linear');
129     effiValu = interp1(betaVect, ETAsline, BETA, 'linear');
130
131 end
132
133 else
134
135 %% Interpolate Off-Design Turbine Performance
136
137 % Step 1: Find Beta corresponding to required pressure ratio
138 %============================================================
139 BETA = interp1(presRatiRange, betaRange, PR);
140
141 % Check if required pressure ratio is within bounds of map data
142 if isnan(BETA) == 1
143     error('Turbine interpolation unsuccessful, but within ...
            map bounds')
```

```
144 end
145
146 % Step 2: Find Pressure Ratio at current BETA value
147 %============================================================
148
149 % Interpolate to Find Mass Flow
150 massFlowValu    = interp2(speeVect, betaVect, ...
        massFlowTabl',            ...
151                 N, BETA);
152
153 % Interpolate to Find Efficiency
154 effiValu        = interp2(speeVect, betaVect, effiTabl', ...
                      ...
155                 N, BETA);
156
157 end
158 return
```

D.2.5 Validation

A similar approach, to that used in validating the compressor map handling, has been adopted for validating the turbine map handling routines developed thus far. In doing so, validation tests were performed at both static sea-level and cruise conditions, for both steady-state and transient operation.

Steady-State, Static Sea-Level Conditions

As with the compressors, the turbine performance map handling was first validated across its running line. The running line was obtained from GSP according to the steady-state running line of the engine. The range of the running line used for the compressor and turbine maps do not correspond directly, and a larger engine running line range was used for the turbine validation. This is because the envelope of the compressor maps could be reached quite easily with a smaller running line range of the engine. Whereas, the turbines are able to accommodate a far more extensive running line. In doing these validations, the largest possible range permitted by GSP was used. The resultant turbine running lines at steady-state static sea-level conditions, are depicted in Figures D.32a and D.32a, and the corresponding errors between GSP and the model developed are given by D.32b and D.32b.

As with the compressor handling errors, it is observed that the smallest error occurs at the design point and is amplified as the running line moves away from this point. This error, as before, is attributed to discrepancies in scaling through the introduction of rounding errors from GSP's output. Other than that, the errors generally fall well within 1% and are deemed small enough to validate the model.

It was difficult to test the turbine map handling procedures outside of the map bounds. This is because very low and very large fuel flow caused

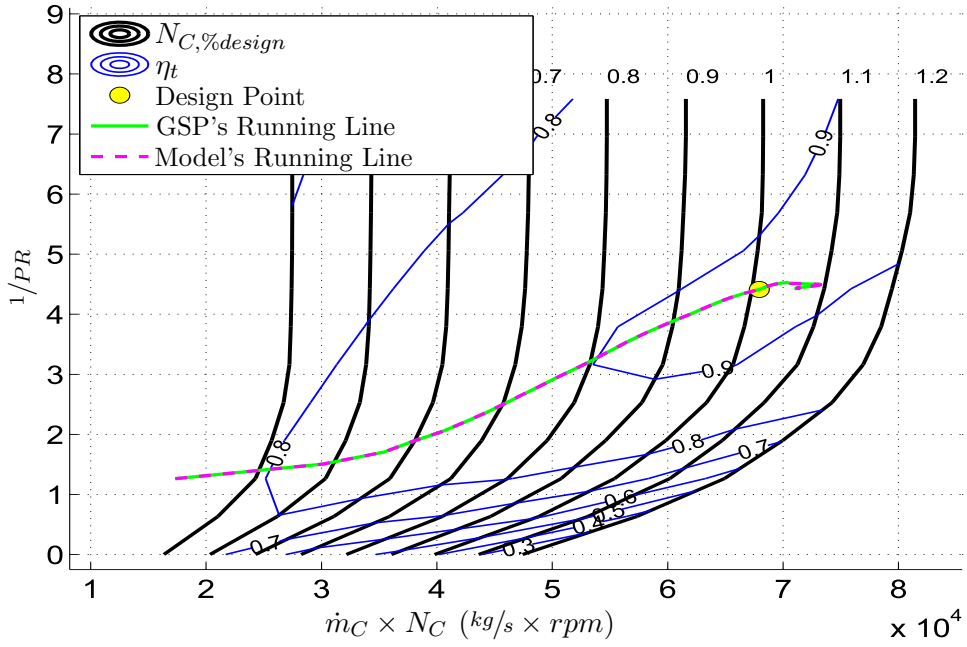
convergence issues in GSP's combustion model. Therefore, extrapolation was encountered only in the low pressure turbine, Figure D.32a, at low speeds. Nonetheless, this extrapolation yielded sufficiently accurate results as is evident from Figure D.32b.

In light of the above, it was concluded that the turbine map handling model developed, produced sufficiently accurate results. However, as with the compressors, it is important to validate the model's transient capabilities at cruise conditions.

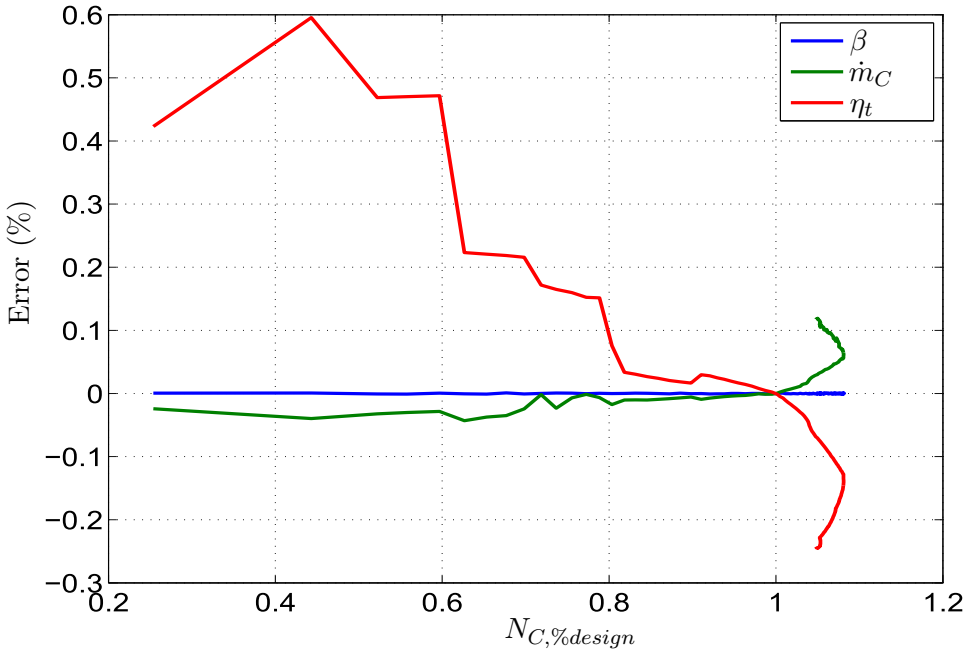
Transient Cruise Conditions

As with the compressor map validation, the turbine map handling model was tested under extreme transients at cruise. A different transient schedule, to that used in validating the compressor map handling model, was required to try and push the turbine's operating trajectories as far from the steady-state running line as possible. This was necessary to test for any vulnerabilities in the routines developed. A transient schedule was found via trial and error, by altering the fuel schedule to GSP's engine model. The schedule used, was found to be at the operating limits of the engine, which was limited by the capabilities of GSP's combustion model as well as the operating limits of the compressors. The resulting transient trajectories ar depicted in Figures D.34a and D.35a, with corresponding errors in Figures D.34b and D.35b.

In both cases, the transient schedule was able to exceed the upper speed limit of the turbines. Despite using extrapolation, an error within 0.5% was still achieved. In general, the error, on both maps, were small enough not to be of any concern. In addition, there was no evidence of any strange anomalies occurring. It was concluded that because the turbines were tested at the engine's operational limits, and because errors seldom exceed 1%, that the turbine map handling was correct and valid.

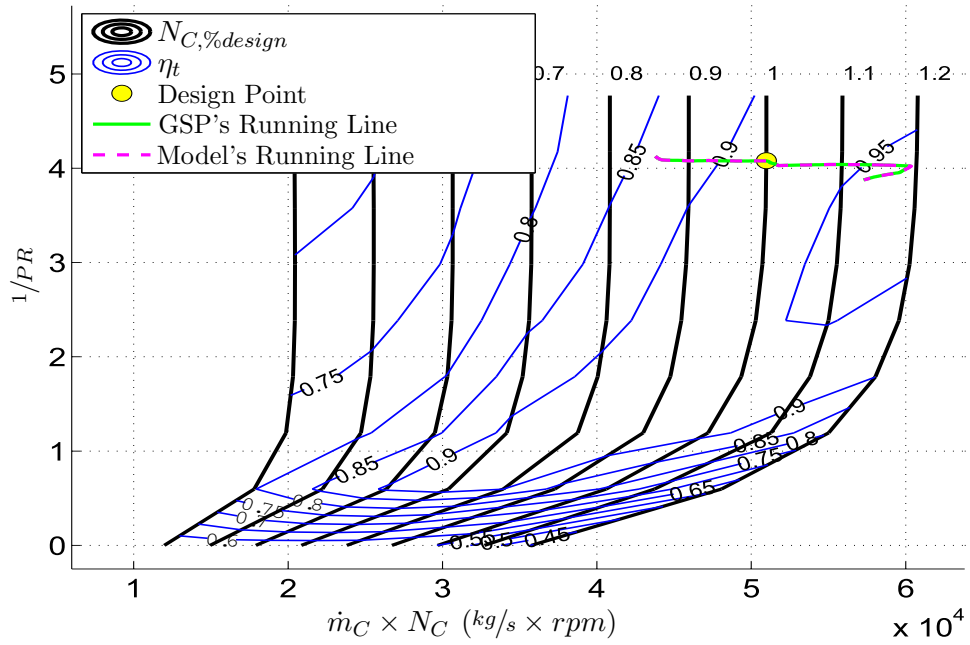


(a) Turbine map with running line

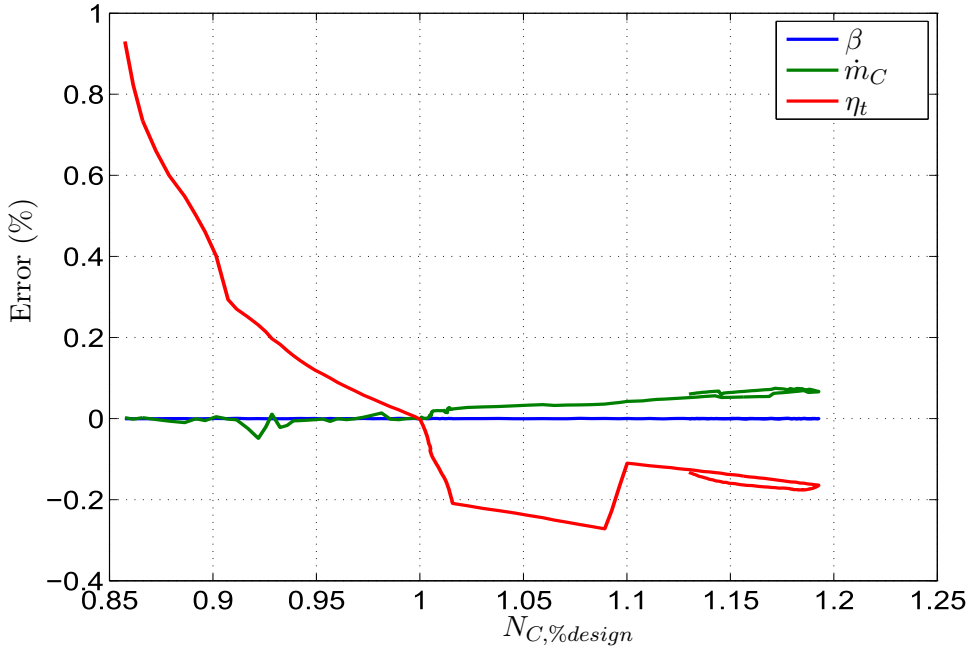


(b) Corresponding interpolation error

Figure D.32: Low pressure turbine map interpolation: Steady-state running line at static sea-level conditions

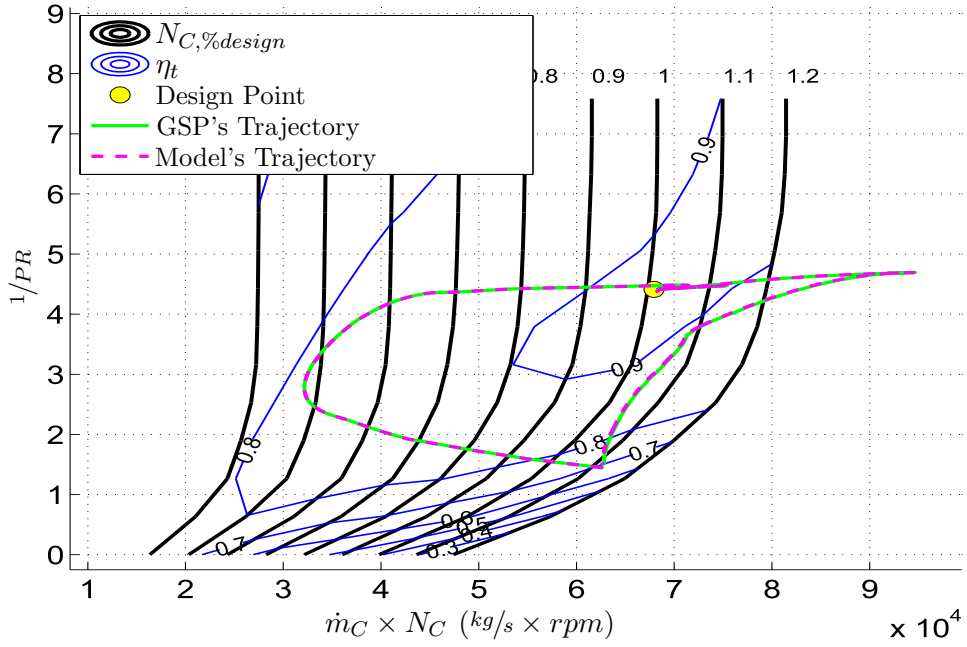


(a) Turbine map with running line

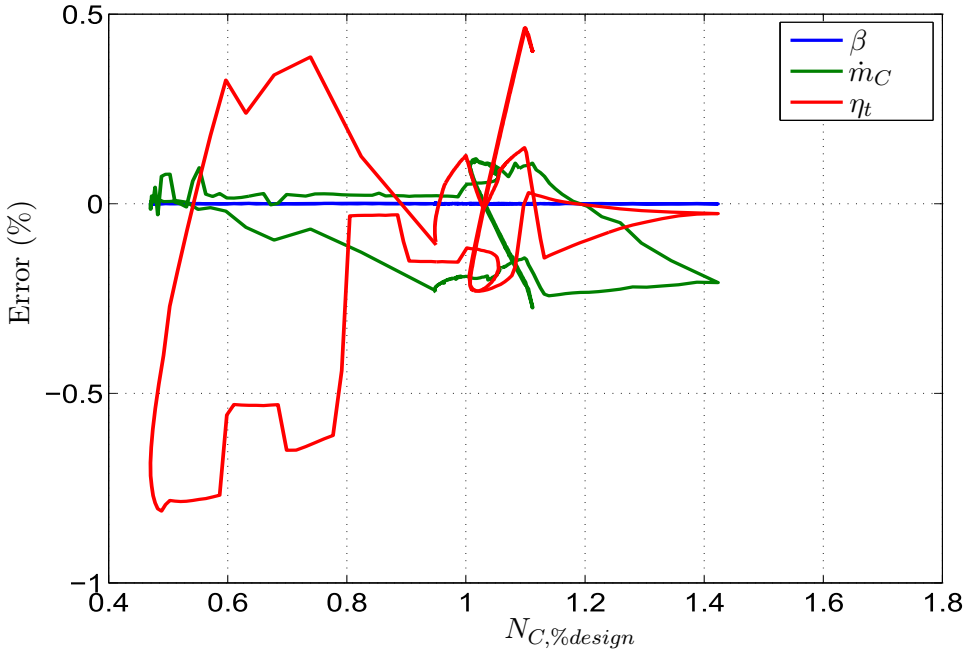


(b) Corresponding interpolation error

Figure D.33: HighNC pressure turbine map interpolation: Steady-state running line at static sea-level conditions

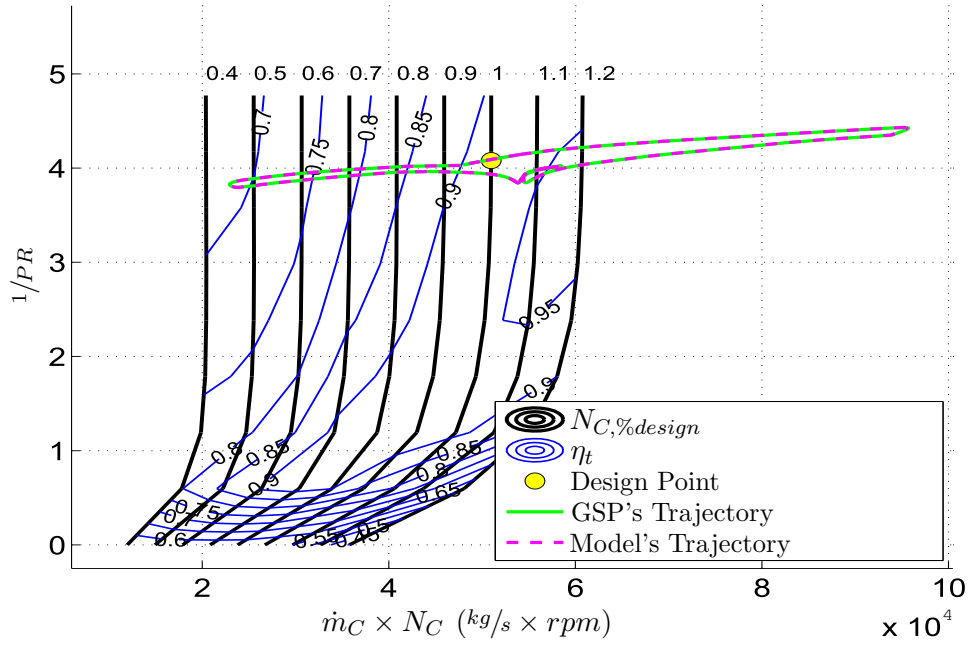


(a) Turbine map with transient trajectory

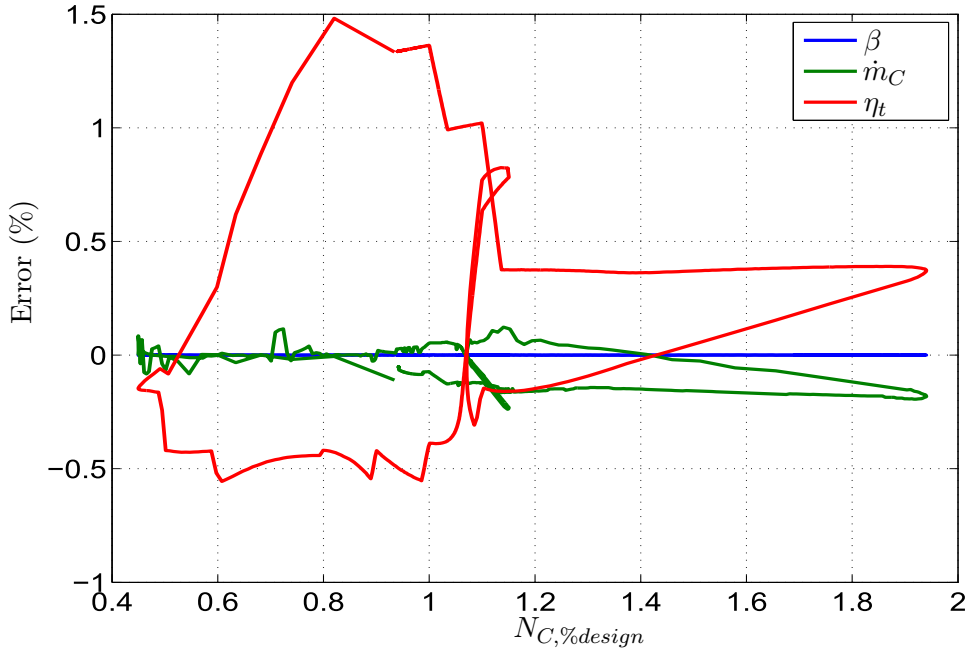


(b) Corresponding interpolation error

Figure D.34: Low pressure turbine map interpolation: Transient trajectory at cruise conditions



(a) Turbine map with transient trajectory



(b) Corresponding interpolation error

Figure D.35: High pressure turbine map interpolation: Transient trajectory at cruise conditions

Appendix E

Engine Component Modelling

There are a number of texts which describe the aerothermodynamics of gas turbine engines. Saravanamuttoo *et. al.* [128] is the most noteworthy and cited of such texts, however Flack [33] provides a well structured text with a detailed and comprehensive description of each component. Therefore, the theory presented by Flack [33] forms the basis of the thermodynamic theory used in this work, and is summarised in the sections that follow. Other sources are referred to where necessary.

The operation of the entire engine is defined by its components as well as their layout and interactions with one another. Therefore, the engine may be modelled mathematically by combining the mathematical descriptions of each component. This chapter describes the mathematical modelling of the thermodynamic operation of each individual component, i.e. intake, fan, compressors, combustor, turbines and nozzles.

E.1 Flight Conditions

Before the component models are discussed, it is important to take a look at the environment, in which it is submersed, and therefore this section describes the modelling of the atmosphere for different flight conditions. A brief description of the theory is presented, followed by details of the Simulink block developed to model this theory. Finally, the model is validated by comparison against GSP software.

E.1.1 Theory

The environmental conditions, surrounding and entering the engine, are simply determined according to the altitude and airspeed of the aircraft. There are many different models available to determine the change in atmospheric

conditions with altitude. This work makes use of the International Standard Atmosphere (ISA) model [129].

Typically, one may obtain static temperature, static pressure, local speed of sound and density for a given altitude, from the ISA model. However, because the aircraft is in motion, relative to the the atmosphere, it is more convenient to use the total/stagnation temperatures and pressures associated with the aircraft's airspeed. The total temperature and pressure are obtained through Equations E.1 and E.2, respectively.

$$T_t = T_a \left[1 + \frac{\gamma - 1}{2} M^2\right] \tag{E.1}$$

$$P_t = P_a \left[1 + \frac{\gamma - 1}{2} M^2\right]^{\frac{\gamma}{\gamma - 1}} \tag{E.2}$$

However, the specific heats ratio, found in Equations E.1 and E.2, is a function of total temperature and pressure. Therefore, an iterative procedure is required to simultaneously determine the specific heats ratio, total temperature and total pressure. As a simplification, it is assumed that specific heats ratio is a function of temperature only, which has the more dominant effect on specific heats, than pressure.

A flow diagram of the iterative procedure, used to calculate the total atmospheric temperature an pressure, relative to the aircraft motion, is depicted in Figure E.1. First, the total temperature is estimated to equal the atmosphere's static temperature, from which the corresponding ratio of specific heats is determined. A new estimated of the total temperature is calculated, based on this newly estimated specific heats ratio. The solution converges if the comparison of the new and old estimates falls within a tolerance defined by the user. If not, the new estimate becomes the old, and is used to re-estimate the specific heats ratio, which is in-turn used to calculate a new total temperature estimate. This process is repeated until the required tolerance is met. The resulting total atmospheric temperature is then used to determine the remaining thermodynamic properties of the air, as well as the total pressure.

E.1.2 Simulink Model

The Simulink block, developed to model the flight conditions with respect to the engine, is depicted in Figure E.2. Figure E.2a, illustrates the highest level block, which is used for interfacing with other engine components. Attached to this block are the inputs of altitude and airspeed, which are used to determine the block's outputs. There are two output ports, the first containing a bus signal of all the properties calculated (see Figure E.2c) and a second the static pressure. Although the static pressure is contained within the bus signal from the first output, it is required as an input to other

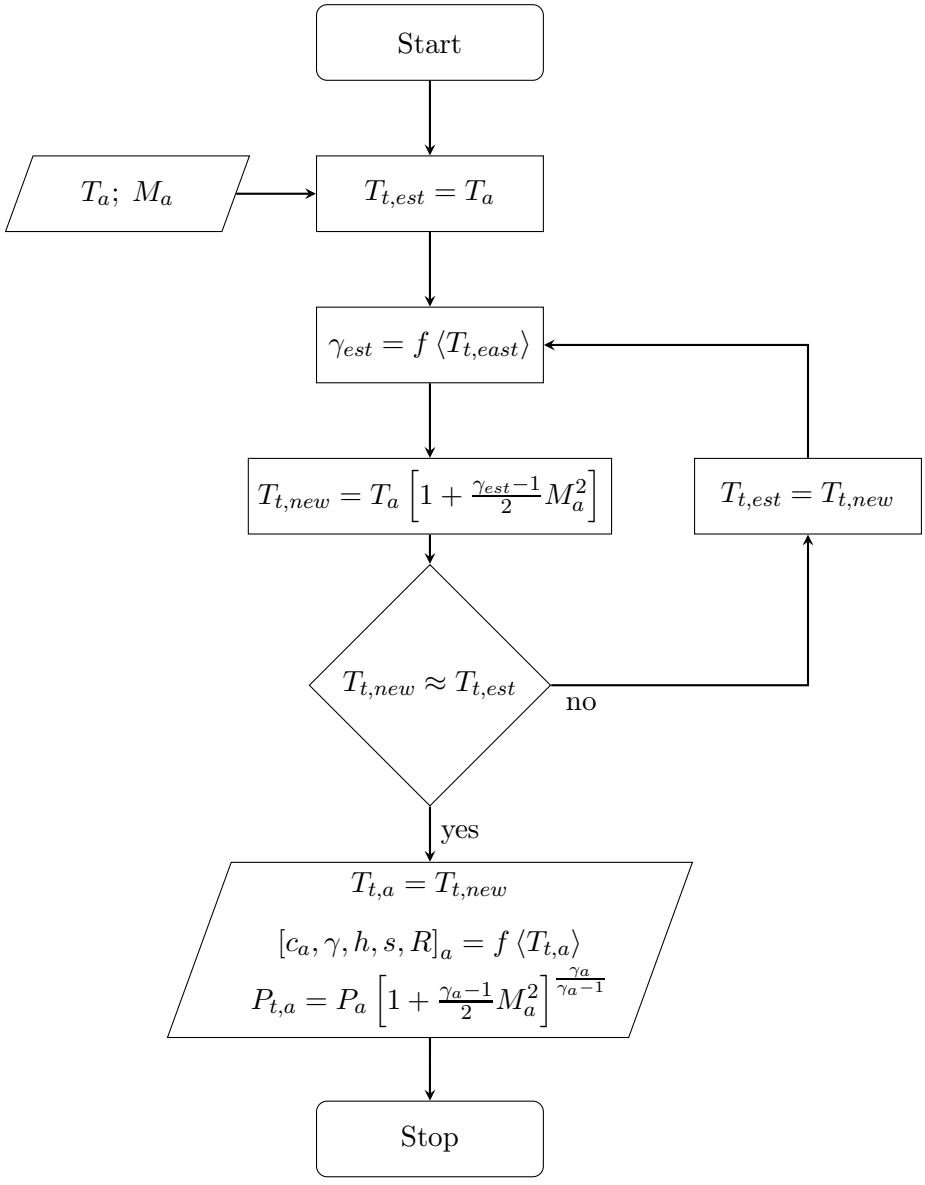


Figure E.1: Flow chart: The atmospheric total temperature is calculated iteratively to account for airspeed

engine components (e.g. nozzles for thrust calculations), and therefore this port is purely for convenience when assembling an engine model.

Figure E.2b, illustrates the subsystem "Flight Conditions" of Figure E.2a. This subsystem is used for interfacing with the user. The user may define the name of the vector, containing recordings of the block's outputs, which is then written to the MATLAB workspace upon completion of the simulation. This allows for convenient access of the output data, post simulation. This level may seem unnecessary, but it serves as a temporary replacement for a subsystem "mask", and this architecture will persist with the component models that follow. The idea is to hide the details of the model from the user, while still allowing the user to define the necessary parameters. "Masks" have not yet been created for the engine model in this work, and are left for future work.

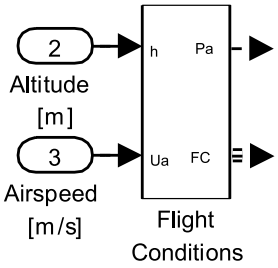
Figure E.2c, illustrates the "Flight Conditions Calculator" subsystem of Figure E.2b. This subsystem contains the details of the model's calculations. The first subsystem, "ISA Atmosphere Model", is an ISA model obtained from Simulink's Aerospace blockset. This subsystem determines atmospheric static temperature, static pressure, speed of sound and density for a given altitude. The Mach number is then calculated from the airspeed, of the aircraft, and the speed of sound determined from the ISA subsystem; Equation E.3.

$$M_a = \frac{u_a}{a_a} \tag{E.3}$$

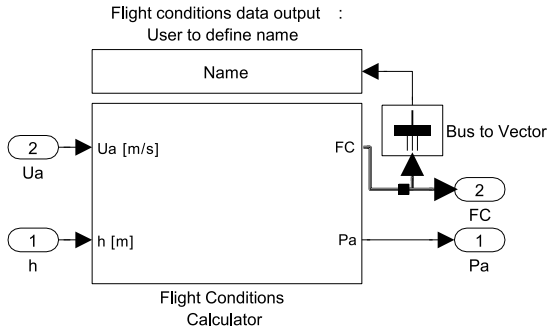
An Embedded MATLAB Function, "FCTHERM", is then used to calculated total conditions according to the procedure detailed in Figure E.1. As indicated, "FCTHERM" takes in the specified tolerance, static temperature, Mach number and static pressure, respectively, and returns the total temperature, total pressure, specific heat, specific heat ratio, enthalpy, entropy and gas constant. The outputs from FCTHERM are combined with the outputs from the ISA model. A bus signal is used to pass these parameters to the higher levels of the block and on to other engine components.

E.1.3 MATLAB Code

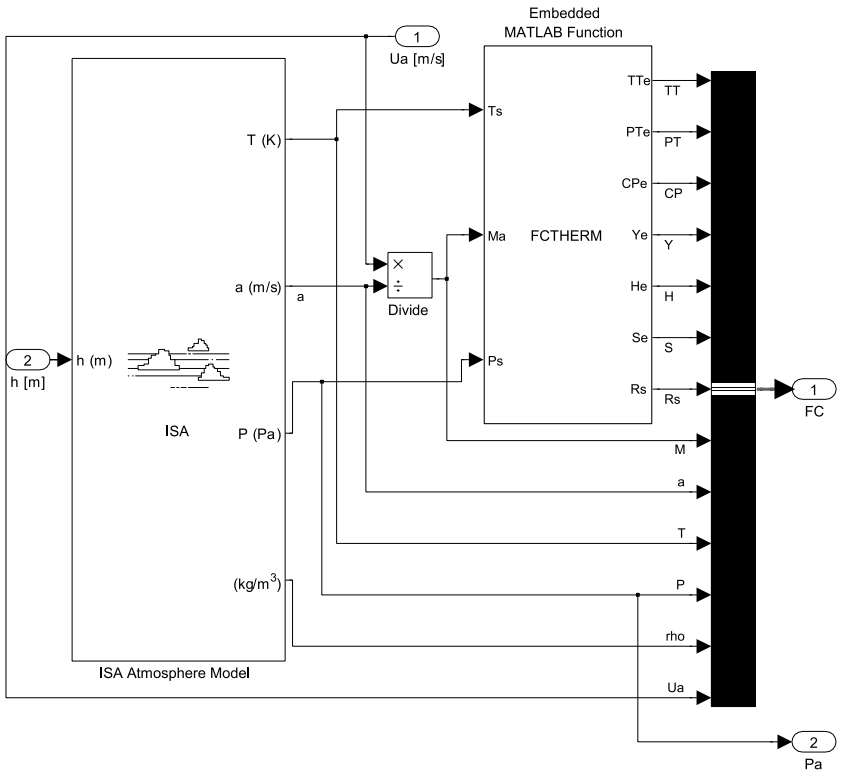
This subsection lists the MATLAB code called by the flight conditions block. The Embedded MATLAB Function, "FCTHERM" of Figure E.2c, is used to call a MATLAB function 'FCTherm.m'. FCTHERM and FCTherm.m's details are presented in Code Listings E.1 and E.2, respectively. FCTherm.m, represents the code that performs the procedure illustrated by the flow chart of Figure E.1, and the reader is kindly requested to refer to the code's comments for explanation where necessary. This method of calling a MATLAB function m-file from and embedded function, allows for iterative procedure to take place per time step of the simulation. 'FCTherm.m' calls one sub-



(a) Level for interfacing with other components



(b) Level where user may define tolerance for temperature calculations, as well as a data set name



(c) Flight condition block subsubsystem: Flight conditions calculator

Figure E.2: Simulink block to determine the atmospheric conditions entering the engine

routine 'AirTherm.m', which returns the thermodynamic properties of air at a given temperature. 'AirTherm.m' is described in detail in Appendix C.

Code Listing E.1: Embedded MATLAB Function FCTHERM

```
function [TTe, PTe, CPe, Ye, He, Se, Rs] = FCTHERM (Ts, Ma, Ps)

% MATLAB function to be called
eml.extrinsic('FCTherm');

% Initialise output variables
TTe = 0; PTe = 0; CPe = 0; Ye = 0; He = 0; Se = 0; Rs = 0;

% Call MATLAB function to calculate outputs
[TTe, PTe, CPe, Ye, He, Se, Rs] = FCTherm (Ts, Ma, Ps);
```

Code Listing E.2: MATLAB Function FCTherm.m

```
%============================================================
%       FLIGHT CONDITIONS THERMODYNAMIC CALCULATIONS
%============================================================
%
% NAME:         FCTherm.m
%
% AUTHOR:       D. Sanders, University of Cape Town
%
% DATE:         17 July 2012
%
% FUNCTION:     [TT, PT, CP, Y, H, S, Rs] = FCTherm (Ttol, ...
%               Ts, Ma, Ps)
%
% PURPOSE:      Calculate atmospheric thermodynamic properties
%
% INPUTS:
%   Ts          = Static Temperature                      [K]
%   Ma          = Mach number                             [-]
%   Ps          = Static Pressure                        [Pa]
%
% OUTPUTS:
%   TT          = Total Temperature                       [K]
%   PT          = Total Pressure                         [Pa]
%   CP          = Specific heats                     [kJ/kg-K]
%   Y           = Specific heats ratio                    [-]
%   H           = Enthalpy                            [kJ/kg]
%   S           = Entropy                            [kJ/kg-K]
%   Rs          = Gas constant                       [kJ/kg-K]
%
% SUBROUTINES:  AirTherm.m
%
% REFERENCES:
%
```

```
34 %   Flack, R. Fundamentals of jet propulsion with applications
35 %   Cambridge Univ Pr, 2005
36 %
37 % For coding conventions, please refer to:
38 %
39 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
40 %   ver 1.5. October 2002.
41 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
42 %=============================================================
43
44 %% IntakeTherm.m function:
45
46 function [TT, PT, CP, Y, H, S, Rs] = FCTherm (Ts, Ma, Ps)
47
48 %% Thermodynamic Calculations:
49
50 % Initialise estimates of total temp and specific heats ratio
51 TTest = Ts;
52 TTnew = TTest;
53
54 % Inititalise while loop
55 deltTT = 1000;
56
57 % Tolerance of temp calculation
58 Ttol = 0.0001; % [K]
59
60 % Calculate Tte using cp, y at average component temp
61 while deltTT > Ttol
62
63     % New estimate of gas properties
64     [CPest, Yest, Hest, Sest, Rsest] = AirTherm (TTnew);
65
66     % Calculate Total Temperature
67     TTnew = Ts * (1 + ( (Yest-1)/2 * Ma^2 ));
68
69     % Calculate differnce between old and new estimates
70     deltTT = abs(TTnew - TTest);
71
72     % Re-estimate
73     TTest = TTnew;
74 end
75
76 % Actual TT equals most recent estimate
77 TT = TTnew;
78
79 % Use final total temp to calculate gas properties
80 [CP, Y, H, S, Rs] = AirTherm (TT);
81
82 % Calculate total pressure
83 PT = Ps * (1 + ( (Y-1)/2 * Ma^2 ))^(Y / (Y - 1));
84
85 return
```

	Static sea-level			Cruise		
Parameter	**GSP**	**FCB**	**Error %**	**GSP**	**FCB**	**Error %**
Alt (m)	0	0	0	12192	12192	0
Air Spd $(^m/_s)$	0	0	0	236.100	236.100	0
T_t (K)	288.150	288.150	0	244.390	244.450	2.455E-02
P_t (kPa)	101.325	101.325	0	28.590	28.590	0
c_p $(^J/_{kg}K)$	1004.280	1004.280	0	1002.970	1002.970	0
γ $(-)$	1.400	1.400	0	1.401	1.401	7.138E-05
h $(^{kJ}/_{kg}K)$	-14.379	-14.379	0	-58.297	-58.236	1.046E-01
s $(^J/_{kg}K)$	6829.920	6829.930	1.464E-04	7027.810	6664.872	5.164E+00
R $(^J/_{kg}K)$	287.050	287.100	1.742E-02	287.050	287.100	1.742E-02
M $(-)$	0.000	0.000	0	0.800	0.800	7.874E-03
a $(^m/_s)$	340.320	340.294	7.502E-03	295.093	295.070	7.895E-03
ρ $(^{kg}/_{m^3})$	1.225	1.225	0	0.302	0.302	0

Table E.1: Validation of the flight conditions block against GSP software

E.1.4 Model Validation

The flight conditions model has been validated against the GSP's atmospheric model, for static sea-level and cruise conditions (Altitude of 12 192 m ans airspeed of 236.1 m/s). As is evident from Table E.1, there is an excellent correlation between the two models; i.e the flight conditions block (FCB) developed in this work, and the atmospheric model in GSP software. All parameters, except one, are found to deviate by less than a percent from those calculated by GSP software. The exception, entropy at cruise condition, deviates by approximately 5%. This is attributed to the possibility that GSP makes use of a more sophisticated model, which accounts for the influence of pressure on gas thermodynamic properties. Nevertheless, it is noted that entropy is not used in any of the engine calculations, and therefore this error does not effect the overall accuracy of the engine model. In addition, during the comparison, the results obtained from the FCB were rounded to match the number of significant figures outputted by GSP for each of the parameters. Therefore, some of the deviations are also attributed to possible rounding errors.

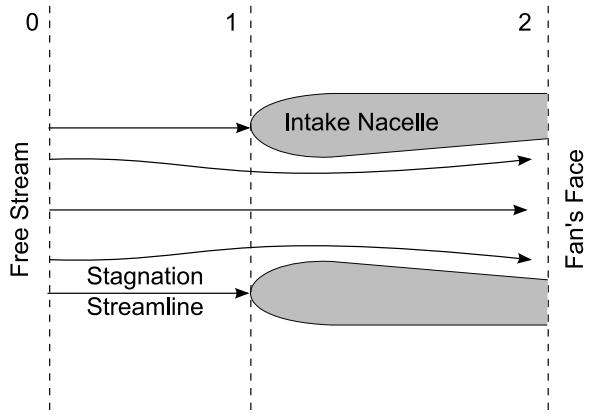


Figure E.3: Schematic of the intake with station numbering

E.2 Intake

This section discusses the intake model incorporated within the engine model. First, a theoretical look at the intake process is discussed, followed by a description of the MATLAB/Simulink model developed. Finally, the MATLAB/Simulink model is validated against GSP's intake model.

E.2.1 Theory

A schematic of the intake is depicted in Figure E.3, and indicates the station numbering used with reference to the entire engine. The intake process begins in front of the engine and ends at the inlet to the fan. The purpose of the intake process is to deliver the free-stream air, at a reduced velocity and increased static pressure, to the fan's inlet. Seddon and Goldsmith [118] give an extensive description on the operation of an intake, and forms the basis of the intake theory covered here.

The efficiency of the intake process is commonly referred to as the intake "pressure recovery", and is simply the ratio of fan inlet total pressure over that of the free-stream condition, i.e. the pressure ratio across the intake:

$$PR_{intake} = \frac{P_{t,2}}{P_{t,0}} = \frac{P_{t,2}}{P_{t,1}} \frac{P_{t,1}}{P_{t,0}} \tag{E.4}$$

As indicated by Equation E.4, the pressure recovery occurs in two stages; from the free-stream to the face of the intake, and from the intake face to the face of the fan. Different forms of pressure losses may occur during either of these stages, however the main contribution to pressure loss is located between stations 1 and 2.

The pressure recovery, between stations 1 and 2, is effected by pressure losses as a result of skin friction and flow separation. However, when it occurs, the pressure losses caused by flow separation more than offset those

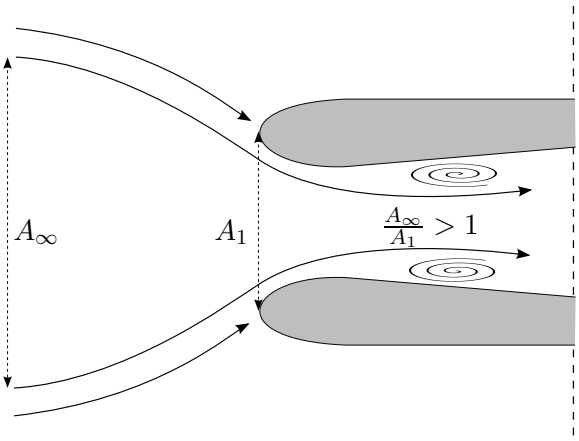


Figure E.4: Intake flow ratio, depicting separation conditions

resulting from skin friction. There are a number of factors which may cause flow separation. If the free-stream area "captured" by the intake (A_∞) is significantly larger than that of the intake (A_1), the stagnation point will shift towards the outside of the nacelle, thus increasing flow velocity around the intake's lip to a point where separation occurs. Flow separation is likely when $\frac{A_\infty}{A_1}$, referred to as the "flow ratio", exceeds unity, as depicted in Figure E.4.

Furthermore, at high angles of attack (or in extreme gusts), the air will enter at a skewed path, which is likely to cause asymmetric flow separation if the angle of attack exceeds 10°; as depicted in Figure E.5.

Intake flow separation results in distortion (non-uniformities in pressure and temperature distributions) of the flow delivered to the compressors. Both radial and circumferential distortion may occur. This distortion negatively effects compressor operation, and may be interpreted as a shifting of the surge line to a more restrictive position. However, because aircraft fly at such high speeds when close to cruise, it is assumed that lateral/vertical atmospheric velocities (due to atmospheric turbulence or induced by a nearby vortex of another aircraft) will be extremely small relative to the aircraft's airspeed. Therefore, it is assumed that the afore mentioned conditions, that cause flow separation will, not occur.

Furthermore, the intake is assumed to deliver air to the fan uniformly. The inclusion of non-uniformities is very complex and computationally demanding, requiring high fidelity compressor models. Therefore, it is assumed that distortion conditions won't be encountered, thus allowing for their effects on engine performance to be omitted.

In addition, it is common to assume a pressure recovery of unity for aircraft in subsonic flight. Although commercial aircraft fly in a transonic region, subsonic conditions are assumed throughout this work for simplifica-

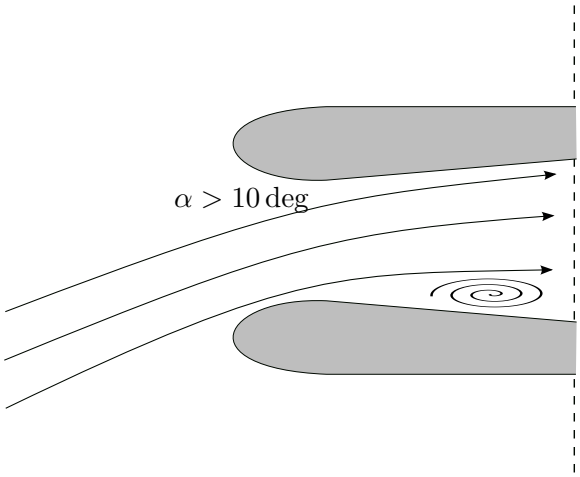


Figure E.5: Schematic illustrating asymmetric flow separation inside the intake, due to a large angle of attack

tion. It is also assumed that the intake operates adiabatically, and therefore the intake's exit total pressure and temperature is equal to that of the free-stream condition. Based on these assumptions, the exit total pressure and temperature is given by Equations E.5 and E.6, respectively.

$$P_{t,2} = P_{t,0} = P_0 \left[1 + \frac{\gamma - 1}{2} M_0^2\right]^{\frac{\gamma}{\gamma - 1}} \tag{E.5}$$

$$T_{t,2} = T_{t,0} = T_0 \left[1 + \frac{\gamma - 1}{2} M_0^2\right] \tag{E.6}$$

Equations E.5 and E.6 hold true for conditions where there is flow compatibility between the intake and the fan. However, during some off-design steady conditions, and certainly unsteady conditions, the fan will demand either a greater/lesser mass flow rate than is delivered by the intake. This may be accounted for by considering the intake's operation in terms of conservation of mass, energy and momentum, where the fluid is assumed to be inviscid but compressible. However, a simpler approach is to assume incompressible flow and therefore continuity is maintained through the automatic, instantaneous adjustment of the free-stream "capture" area. Therefore, this assumptions justifies the use of Equations E.5 and E.6.

Finally, it is assumed the air ingested comes to a halt at the face of the fan. This is because, during the diffusion process, the kinetic energy of the ingested flow has been transferred to potential energy in the form of an increase in static temperature and pressure. This results in a force being applied to the engine, which is referred to as ram drag. The ram drag is simply equal to the change in momentum of the fluid, Equation E.7, and opposes the thrust forces produced at the primary and secondary nozzles.

$$D_{ram} = u_a \times \dot{m}_{intake} \tag{E.7}$$

E.2.2 Simulink Model

Figure E.6, illustrates the intake model that has been developed in Simulink. Figure E.6a, illustrates the highest level of the intake model, which is used for interfacing with other engine components and atmospheric conditions. The model takes in the outputs of the flight conditions block and the mass flow rate 'dictated' by the fan downstream. The block outputs the thermodynamic properties of the gas exiting the intake; total temperature, total pressure, specific heat, enthalpy, entropy and gas constant, respectively, as well as the ram drag experienced. It must be noted, however, that unlike the output of the flight conditions block, the Mach number, local speed of sound and density have been omitted. This is because these parameters are not necessary to define the operation for the downstream components. A higher fidelity engine model would require such parameters to be passed between components. In addition, there is an output port fro the ram drag for convenience when assembling an engine model, as the ram drag must be combine with the thrust produced at each nozzle to obtain a net thrust.

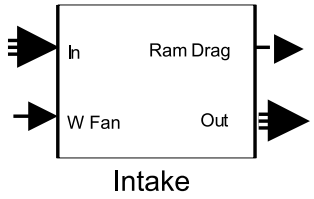
Figure E.6b, depicts the user-interface level, where the user is able to define the pre-entry and intake pressure losses, which are, thereafter, held constant. At this level, as before, the user is able to allocate a name (usually a station number) to the vector of gas properties, exiting the intake, for outputting to the MATLAB workspace. This name allows for easy access to the data recorded during a simulation. Figure E.6c, illustrates the contents of the "Difussion" subsystem, which represents Equations E.5, E.6 and E.7.

E.2.3 MATLAB Code

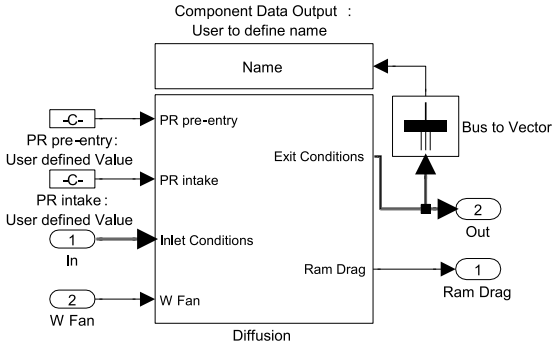
No additional MATLAB code was required for this component model, as the theory is self-contained within the Simulink blocks developed.

E.2.4 Validation

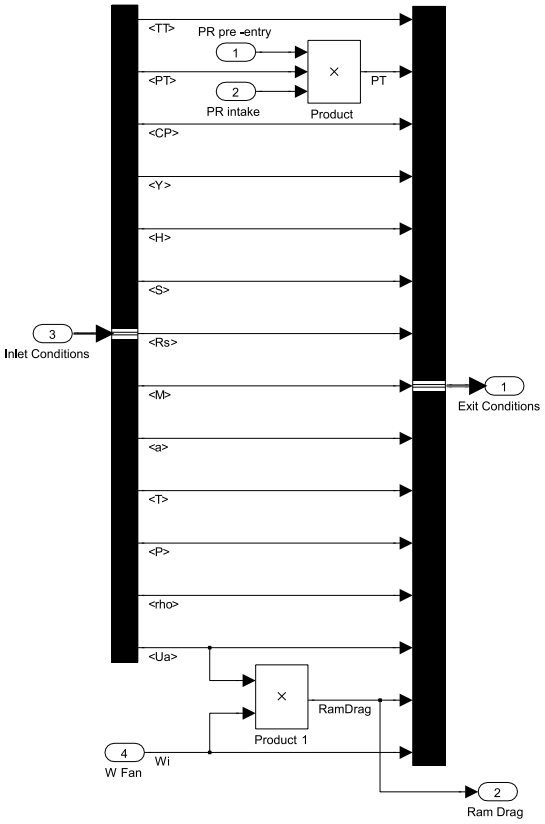
Little validation of this component model was required. As presented in the theory section, GSP too uses a user defined pressure recovery value which is held constant. Therefore, all the flight condition parameters, except total pressure ratio, remain unchanged across the intake. GSP also assumes a pressure recovery of 1 for this type of engine, therefore the total pressure also remains unchanged. In addition, the calculation of ram drag is more dependent on the the accuracy of the mass flow rate determined by the fan component (the airspeed is specified directly and is therefore not of concern), and at this point does not require validation.



(a) High-level intake block



(b) Intake subsystem for user defined parameters



(c) Diffusion subsystem

Figure E.6: Intake model developed in Simulink

Therefore, the only form of validation required was to check if GSP included any transient effects across the intake. A transient simulation in GSP, for a slam acceleration, revealed that the the stagnation values for the thermodynamic parameters remained constant across the intake. A step change in fuel was applied to the entire engine in GSP, shifting from a fuel input of 1 kg/s to 3.5 kg/s in 0.1 seconds. The results of this simulation are tabulated in Table E.2. Typically an engines response time is around 6-10 seconds. The results presented in Table E.2, reveal that no change in thermodynamic properties occurs across the GSP engine's intake during a transient, thereby validating the intake model developed in this section.

Time	$\dot{m}_f$	$T_{t,1}$	$T_{t,1}$	$P_{t,1}$	$P_{t,2}$	γ_1	γ_2
(seconds)	$(^{kg}/_{m^3})$	(K)	(K)	(bar)	(bar)	(-)	(-)
0	1	288.15	288.15	1.01325	1.01325	1.40023	1.40023
0.1	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
1	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
2	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
3	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
4	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
5	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
6	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
7	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
8	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
9	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023
10	3.5	288.15	288.15	1.01325	1.01325	1.40023	1.40023

Table E.2: Validation of intake model, indicating that thermodynamic properties remain unchanged across the intake during transient operation (at static sea-level)

E.3 Compressors

E.3.1 Theory

The performance of each compressor is best described by four interdependent parameters; rotational speed, pressure ratio, mass-flow rate and isentropic efficiency. These parameters, and how they relate to one another, are most conveniently represented in the form of compressor performance

maps. Compressor maps, and their usage, have been discussed extensively in Appendix D.1

In most cases, given values/estimates of rotational speed and pressure ratio (or mass-flow ratio) are used to obtain corresponding values of mass-flow rate (or pressure ratio) and isentropic efficiency [1]. The isentropic efficiency of a compressor is a comparison of the ideal versus actual power used to drive the compressor, Equation E.8.

$$\eta_c = \frac{h_{s,te} - h_{ti}}{h_{te} - h_{ti}} \tag{E.8}$$

The temperature ratio across the compressor may be obtained by solving Equation E.8. However, this is not an easy calculation because the exit temperature is unknown and the specific heats, which are a function of temperature and pressure, vary along the path of compression. To simplify this problem, an approximation assumes that specific heats across the are constant throughout the compressor (both axially and radially). The specific heat value, deemed most appropriate, is that corresponding to the average temperature and pressure across the compressor. This has been discussed already in detail in Section 4.2.3. An iterative procedure is required to determine the average temperature, and subsequent specific heat, across the compressor due to the interdependency between the two parameters. A further simplification assumes that the specific heats are dependent only on temperature, which is the dominating parameter of influence. Therefore to aid these calculations, an expression is required that relates the change in temperature to a change in pressure for a given constant specific heat value. Such an expression is obtained from Flack [33], Equation E.9, and relates ratios of temperature, pressure and specific heat.

$$TR_{\,c} = \frac{PR^{\,\frac{\gamma-1}{\gamma}} - 1}{\eta_c} + 1 \tag{E.9}$$

An iterative procedure for solving Equation E.9 is depicted in Figure E.7. Initially, the specific heat ratio is estimated to be equal to that of the inlet condition, and therefore the specific heats ratio is found as a function of the compressor's inlet temperature (determined from gas property tables and curve-fits). Subsequently, this estimate is used in conjunction with the pressure ratio (stipulated by the engine model's state vector) and efficiency (obtained from compressor maps) to determine the temperature across the compressor in accordance with Equation E.9. The result of Equation E.9 is multiplied with the inlet temperature, thus providing an estimate of the exit temperature. The exit temperature estimate, averaged with the inlet

[1] Most commonly for the Inter-component Volume Method for modelling engine dynamics. The Iterative Method may use mass flow rate instead of pressure ratio, or alternatively the β parameter discussed in Appendix D.1

temperature, is then used to re-estimate the specific heats ratio. If the specific heat ratio re-estimation value is approximately equal to its predecessor, within a specified tolerance, then it is concluded that the correct exit temperature has been calculated. Subsequently, the thermodynamic properties exiting the compressor are calculated as a function of this newly established exit temperature. However, if the specific heat ratio re-estimation is not approximately the same as its predecessor, then a new exit temperature is calculated based on the re-estimated specific heats ratio and Equation E.9. This is repeated until defining condition is satisfied.

At this point the gas properties at the compressor's inlet and outlet are fully defined according to it stagnation temperature and pressure; which have been used to determine other thermodynamic properties such as specific heats, enthalpy, entropy and gas constant. Furthermore, the compressor's mass flow rate (as well as operational efficiency and β value) has been established from characteristic maps, Section D.1. The remaining operational properties to be defined are those relating to the power, and torque, required by the compressor to operate at the corresponding pressure ratio and mass flow rate. These are important properties when determining engine rotor dynamics. The power, and subsequent torque, required by the compressor is calculated using Equations E.10 and E.11 respectively. The power is given by the specific work (change in enthalpy across the compressor) multiplied by the amount of gas, per second, to which this work is applied (mass flow rate). The torque is then derived from the power through a relation of the compressor's rotational speed.

$$\mathbb{P} = \Delta h \times \dot{m} \tag{E.10}$$

$$\mathbb{T} = \frac{\mathbb{P} \times 60}{2\pi N} \tag{E.11}$$

In summary, the compressor's exit thermodynamic properties, calculated from the theory above, is used to define the inlet boundary condition of the subsequent downstream component. The calculated power is used in determining torque imbalances between compressors and turbines fixed to a common spool, thus dictating rotor dynamics. Lastly the mass flow rate, determined from compressor maps, is required for determining the rates of change in pressure inside the volumes adjacent to the compressor, which is used in modelling the volume dynamics of the engine.

E.3.2 Simulink Model

The Simulink block developed for modelling compressor components is depicted in Figure E.8. Figure E.8a illustrates the highest level subsystem, which is used for interfacing with other engine component models. This block has input ports that take in the compressor's exit pressure, the speed

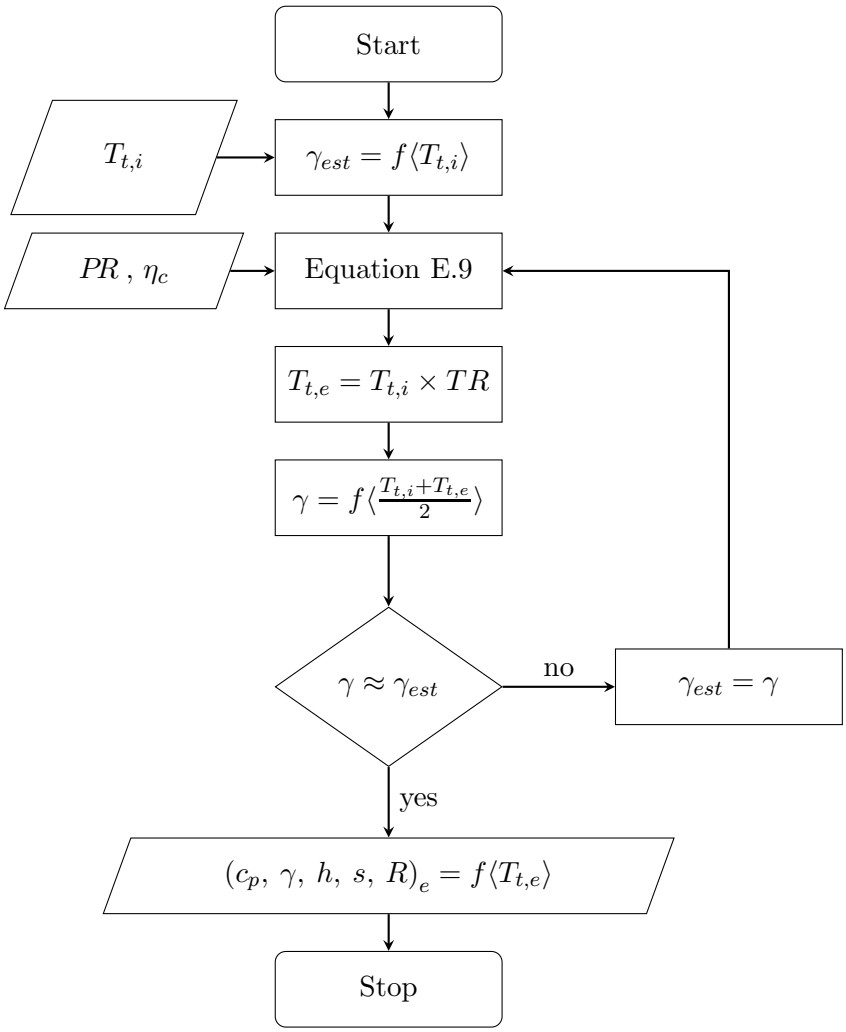


Figure E.7: Iterative procedure for determining the temperature ratio for variable specific heats across a compressor

of the spool to which the compressor is attached, and a bus signal from another component. A new bus signal, containing all of the compressor's associated parameters, is then passed out of a port on the right hand side of the system. This contains information on the compressor map operating point, the thermodynamic state of the gas exiting the compressor and the power/torque required by the compressor. Two additional port, torque and mass flow rate, are included for convenience when integrating with volume and rotor dynamics subsystems of the engine model.

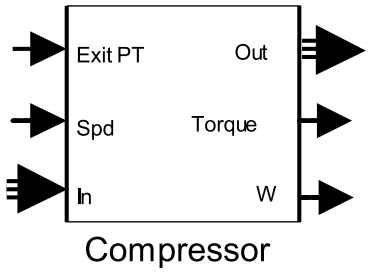
Figure E.8b depicts the user-interface level compressor subsystem. It is here that the user must define the compressor's design point, which is used to scale the generic compressor map [2]. The user is also required to allocate a station number (or name) to the compressor's data output. This name will be used to export all the compressor's data to the MATLAB workspace. The design point is passed to the subsystem 'Compressor Calcs', along with the original signals passed to the high-level block (Figure E.8).

Finally, Figure E.8c illustrates the 'Compressor Calcs' subsystem of Figure E.8b. Here, the signals from the higher levels are past to an 'MATLAB Compressor Function'. The user is required to check that the bus signal selector (black bar on the left hand side) has assigned inlet total pressure and temperature to the correct inlet ports to the MATLAB embedded function. The 'MATLAB Compressor Function' passes information from the Simulink environment to the appropriate MATLAB functions. This allows for complex iterative calculations to be conducted within a single time step of the Simulink environment (The embedded MATLAB function and associated routines are discussed in the next section). As can be seen from the top of Figure E.8c, the previous β value is passed back into the 'MATLAB Compressor Function'. This is used for an initial estimation of the compressor's off-design operating point, as discussed in Section D.1.2. At the start of a simulation β is assigned a value of 0.5, which correlates to the midpoint of the compressor map look up tables. A value of 0.5 provides, statistically, an improved chance that the compressor map interpolation scheme will converge on the correct value (should the look-up table contain non-distinct vectors) than if the initial β value was placed at either end of the speed line (i.e. $\beta = 0$ or $\beta = 1$).

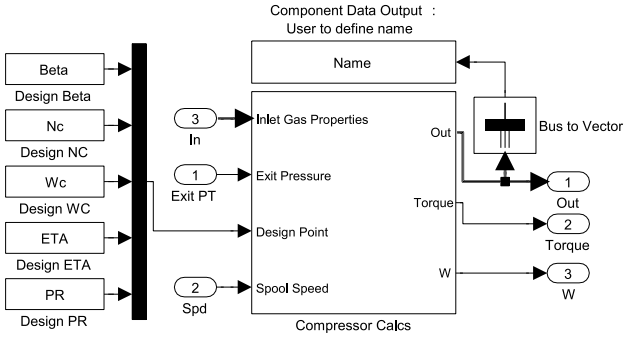
E.3.3 MATLAB Code

The actual compressor calculations are contained within MATLAB functions, and not as Simulink blocks. This is because, as previously mentioned, the compressor calculations require iterative procedures that must be completed within a single time step. This subsection describes the MATLAB functions developed to perform these calculations.

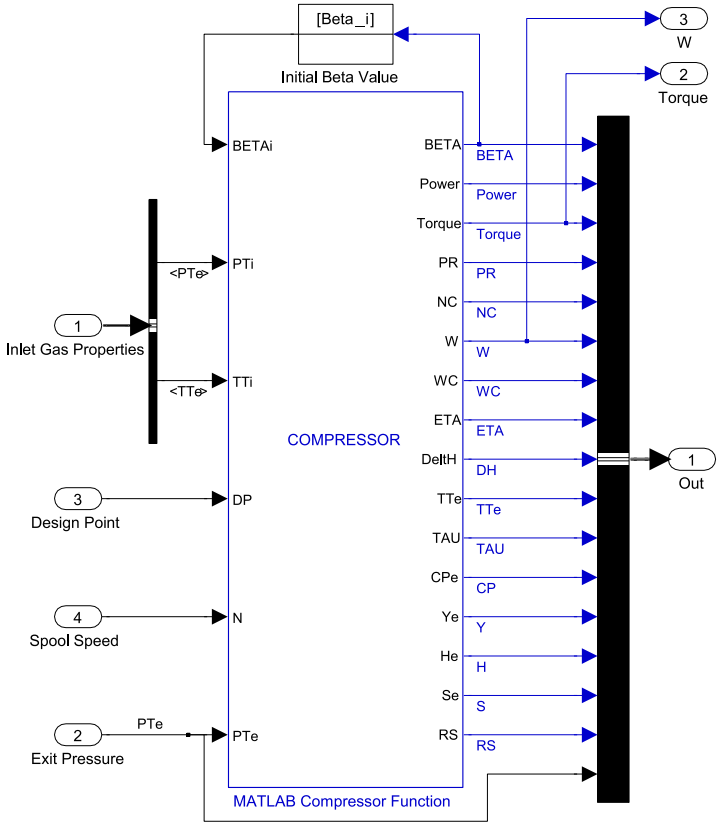
[2] The user is also required to select a general map type suited to high, intermediate or low pressures. This is done so within the Embedded MATLAB Function in Figure E.8c



(a) High-level compressor block



(b) User-interface level



(c) Compressor calculations subsystem

Figure E.8: Compressor model developed in Simulink

The first is an Embedded MATLAB Function, 'MATLAB Compressor Function', who's purpose is solely to facilitate communication of inputs and outputs between the Simulink and MATLAB environments. The contents of this function are presented in Code Listing E.3. It is here that the user is required to specify the map type to be used. An appropriate map may be selected for a low, intermediate or high pressure compressor. The main purpose of this embedded function is to call a subroutine, 'Compressor.m' presented in Code Listing E.4, that performs the relevant calculations in the MATLAB environment. This subroutine first calculates the pressure ratio across the compressor as well as the corrected speed according to the inlet total temperature. These two parameters are then passed to another subroutine, 'CompMapsInterpPR.m', which is used to determine the corrected mass flow rate and isentropic efficiency at which the compressor operates [3]. 'Compressor.m', of Code Listing E.4, then calculates the mass flow rate through the compressor from the corrected mass flow rate interpolated from the compressor map in the previous subroutine call. Next, the pressure ratio, inlet total temperature and efficiency are passed to a subroutine 'CompTherm.m' (Code Listing E.5), which implements the iterative procedure detailed in Figure E.7 to calculate the total temperature (and subsequent thermodynamic properties) exiting the compressor. Finally, 'Compressor.m', calculates the power and torque delivered to the compressor. All the above mentioned parameters are then returned to the Simulink block through the embedded MATLAB function.

Code Listing E.3: Embedded MATLAB Function COMPRESSOR

```
%================================================================
%              COMPRESSOR EMBEDDED MATLAB FUNCTION
%================================================================
%
% NAME:          COMPRESSOR
%
% AUTHOR:        D. Sanders, University of Cape Town
%
% DATE:          21 November 2012
%
% FUNCTION:      [BETA, Power, Torque, PR, NC, W, WC, ETA, ...
    DeltH, TTe, TAU, CPe, Ye, He, Se, RS] = COMPRESSOR ...
    (BETAi, PTe, PTi, TTi, DP, N)
%
% PURPOSE:       Call compressor routines
%
% INPUTS:
%
%   PTe          = Exit Pressure                  [Pa]
%   PTi          = Inlet Pressure                 [Pa]
```

[3] This subroutine is detailed in Section D.1.2 and is found in Code Listing D.2

```
19 %   DP          = Design Point Vector   [BETA NC WC ETA PR]
20 %   TTi         = Inlet Temperature             [K]
21 %   N           = Spool Speed                   [rpm]
22 %   BETAi       = Initial BETA estimate         [-]
23 %
24 % OUTPUTS:
25 %
26 %   BETA        = Map BETA value                [-]
27 %   Power       = Required Power                [W]
28 %   Torque      = Required Compressor Torque    [N.m]
29 %   PR          = Pressure Ratio                [PR]
30 %   NC          = Corrected rotational speed    [rpm]
31 %   W           = Mass Flow Rate                [kg/s]
32 %   WC          = Corrected Mass Flow Rate      [kg/s]
33 %   ETA         = Efficiency                    [-]
34 %   DeltH       = Enthalpy rise                 [J/kg]
35 %   TTe         = Exit Temperature              [K]
36 %   CPe         = Exit specific heat            [J/kg-K]
37 %   Ye          = Exit specific heat ratio      [-]
38 %   He          = Exit enthalpy                 [J/kg]
39 %   Se          = Exit entropy                  [J/kg-K]
40 %   Rs          = Specific Gas Constant         [J/kg-K]
41 %
42 % SUBROUTINES:  Compressor.m
43 %
44 % REFERENCES:
45 %
46 % For coding conventions, please refer to:
47 %
48 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
49 %   ver 1.5. October 2002.
50 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
51 %=================================================================
52
53 function [BETA, Power, Torque, PR, NC, W, WC, ETA, DeltH, ...
       TTe, TAU, CPe, Ye, He, Se, RS] = ...
54     COMPRESSOR (BETAi, PTe, PTi, TTi, DP, N)
55
56 %-----------------------------------------------------------------
57 %                       COMPRESSOR MAP SELECTION
58 %-----------------------------------------------------------------
59
60 % Select general map to represent compressor through ...
       appropriate scaling: mapType = 'LPCD' or 'LPCC' or ...
       'IPC' or 'HPC'
61 mapType = 'IPC';
62
63 %-----------------------------------------------------------------
64 %               VARIABLE INITIALISATION AND FUNCTION CALL
65 %-----------------------------------------------------------------
66
67 % MATLAB function to be called
68 eml.extrinsic('Compressor');
69
```

```
70 % Initialise output variables
71 Power=0; Torque=0; PR=0; NC=0; BETA=0; W=0; WC=0; ETA=0; ...
       DeltH=0; TTe=0; TAU=0; CPe=0; Ye=0; He=0; Se=0; RS=0;
72
73 % Call MATLAB function to calculate outputs
74 function [Power, Torque, PR, NC, BETA, W, WC, ETA, DeltH, ...
       TTe, TAU, CPe, Ye, He, Se, RS] = Compressor (mapType, ...
       DP, PTe, N, PTi, TTi, BETAi)
```

Code Listing E.4: MATLAB Function Compressor.m: For calculating compressor thermodynamics

```
1  %=================================================================
2  %                    COMPRESSOR CALCULATIONS
3  %=================================================================
4  %
5  % NAME:          Compressor.m
6  %
7  % AUTHOR:        D. Sanders, University of Cape Town
8  %
9  % DATE:          17 July 2012
10 %
11 % FUNCTION:      [Power, Torque, PR, NC, BETA, W, WC, ETA, ...
       DeltH, TTe, TAU, CPe, Ye, He, Se, RS] = Compressor ...
       (mapType, DP, PTe, N, PTi, TTi, BETAi)
12 %
13 % PURPOSE:       Calculate compressor operating conditions
14 %
15 % INPUTS:
16 %
17 %   mapType     = type of map to be used (string):
18 %                 low intermediate or high
19 %                 'LPCD', 'LPCC', 'IPC' or 'HPC'
20 %   PTe         = Exit Pressure                 [Pa]
21 %   PTi         = Inlet Pressure                [Pa]
22 %   DP          = Design Point Vector   [BETA NC WC ETA PR]
23 %   TTi         = Inlet Temperature             [K]
24 %   N           = Spool Speed                   [rpm]
25 %   BETAi       = Initial BETA estimate         [-]
26 %
27 % OUTPUTS:
28 %
29 %   BETA        = Map BETA value                [-]
30 %   Power       = Required Power                [W]
31 %   Torque      = Required Compressor Torque    [N.m]
32 %   PR          = Pressure Ratio                [PR]
33 %   NC          = Corrected rotational speed    [rpm]
34 %   W           = Mass Flow Rate                [kg/s]
35 %   WC          = Corrected Mass Flow Rate      [kg/s]
36 %   ETA         = Efficiency                    [-]
37 %   DeltH       = Enthalpy rise                 [J/kg]
38 %   TTe         = Exit Temperature              [K]
```

```
39 %   CPe         = Exit specific heat            [J/kg-K]
40 %   Ye          = Exit specific heat ratio      [-]
41 %   He          = Exit enthalpy                 [J/kg]
42 %   Se          = Exit entropy                  [J/kg-K]
43 %   Rs          = Specific Gas Constant         [J/kg-K]
44 %
45 % SUBROUTINES:  CompMapsInterpPR.m and CompTherm.m
46 %
47 % REFERENCES:
48 %
49 % For coding conventions, please refer to:
50 %
51 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
52 %   ver 1.5. October 2002.
53 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
54 %=============================================================
55
56 function [Power, Torque, PR, NC, BETA, W, WC, ETA, DeltH, ...
       TTe, TAU, CPe, Ye, He, Se, RS] = Compressor (mapType, ...
       DP, PTe, N, PTi, TTi, BETAi)
57
58 %% Compressor Map Handling
59
60 % Calculate Pressure Ratio
61 PR = PTe / PTi;
62
63 % Calculate NC
64 NC = N / sqrt(TTi / 288.15);
65
66 % Map Interpolation
67 [BETA WC ETA] = CompMapsInterpPR (NC, PR, BETAi, mapType, DP);
68
69 % Calculate mass flow rate
70 W = WC * (PTi/101.325e3) / sqrt(TTi / 288.15);
71
72 %% Thermodynamic Calculations
73
74 % Thermodynamic Calculations:
75 [DeltH, TTe, TAU, CPe, Ye, He, Se, RS] = CompTherm (PR, ...
       TTi, ETA);
76
77 %% Power & Torque Calculations
78
79 % Power
80 Power = DeltH * W;
81
82 % Torque
83 Torque = (Power * 60 / (2*pi*N));
84
85 return
```

Code Listing E.5: MATLAB Function CompTherm.m: For implementing Figure E.7

```
%=============================================================
%             COMPRESSOR THERMODYNAMIC CALCULATIONS
%=============================================================
%
% NAME:         CompTherm.m
%
% AUTHOR:       D. Sanders, University of Cape Town
%
% DATE:         17 July 2012
%
% FUNCTION:     [DeltH, TTe, TAU, CPe, Ye, He, Se, Rs] = ...
    CompTherm (PR, TTi, ETA)
%
% PURPOSE:      Calculate exit total temperature and ...
    corresponding
% thermodynamic properties
%
% INPUTS:
%
%   TTi         = Inlet Temperature             [K]
%   PR          = Compressor Pressure Ratio     [-]
%   ETA         = Compressor Efficiency         [%/100]
%
% OUTPUTS:
%
%   DeltH       = Enthalpy rise                 [J/kg]
%   TTe         = Exit Temperature              [K]
%   TUA         = Temperature ratio             [-]
%   CPe         = Exit specific heats           [kJ/kg-K]
%   Ye          = Exit specific heats ratio     [-]
%   He          = Exit Enthalpy                 [kJ/kg]
%   Se          = Exit Entropy                  [kJ/kg-K]
%   Rs          = Gas constant                  [kJ/kg-K]
%
% SUBROUTINES:  AirTherm.m
%
% REFERENCES:
%
%   Flack, R. "Fundamentals of jet propulsion with ...
    applications" Cambridge Univ Pr, 2005, 17
%
% For coding conventions, please refer to:
%
%   Johnson, Richard. "MATLAB Programming Style Guidlines".
%   ver 1.5. October 2002.
%   http://www.mathworks.com/matlabcentral/fileexchange/2529
%=============================================================
%% comptherm.m function:

function [DeltH, TTe, TAU, CPe, Ye, He, Se, Rs] = CompTherm ...
    (PR, TTi, ETA)
```

```
48
49
50 %% Thermodynamic Calculations:
51
52 % Specify specific heat ratio accuracy tolerence
53 ytol = 0.001;
54
55 % Initialise specific heats estimate
56 [CP, Y, H, S, Rs] = AirTherm (TTi);
57 deltY = 1;
58
59 % Inlet enthalpy for Work calcs
60 Hi = H;
61
62 % Calculate Tte using cp, y at average component temp
63 while deltY > ytol
64
65     % Calculate Temperature Ratio (Flack Eq's 3.2.8)
66     TAU = ( ((PR)^((Y-1)/Y) - 1)/ETA ) + 1;
67     % TAU = (PR^((Y-1)/(ETA*Y))); (Uses polytropic ...
           efficiency: used when investigating validation errors)
68     TTe = TAU * TTi;
69
70     % Calculate average temperature and subsequent specific ...
           heat ratio
71     TTm = (TTe + TTi)/2;
72     [CPm, Ym, Hm, Sm, Rs] = AirTherm (TTm);
73
74     deltY = abs(Ym - Y);
75     Y = Ym;
76 end
77
78 % Use final specific heat value to calculate exit temperature
79 TAU = ( ((PR)^((Y-1)/Y) - 1)/ETA ) + 1;
80 TTe = TAU * TTi;
81 [CPe, Ye, He, Se, Rs] = AirTherm (TTe);
82
83 % Calculate Energy input to compressor
84 DeltH = (He - Hi);
85
86 return
```

E.3.4 Validation

The validation of the compressor block begins at the subroutines of the lowest level. The map handling routines have already been validated in Section D.1.5, and therefore only the subroutines 'CompTherm.m' (Code Listing E.5) and 'Compressor.m' (Code Listing E.4) need validating. The validation begins with 'CompTherm.m' as it is also a subroutine of 'Compressor.m'.

CompTherm.m

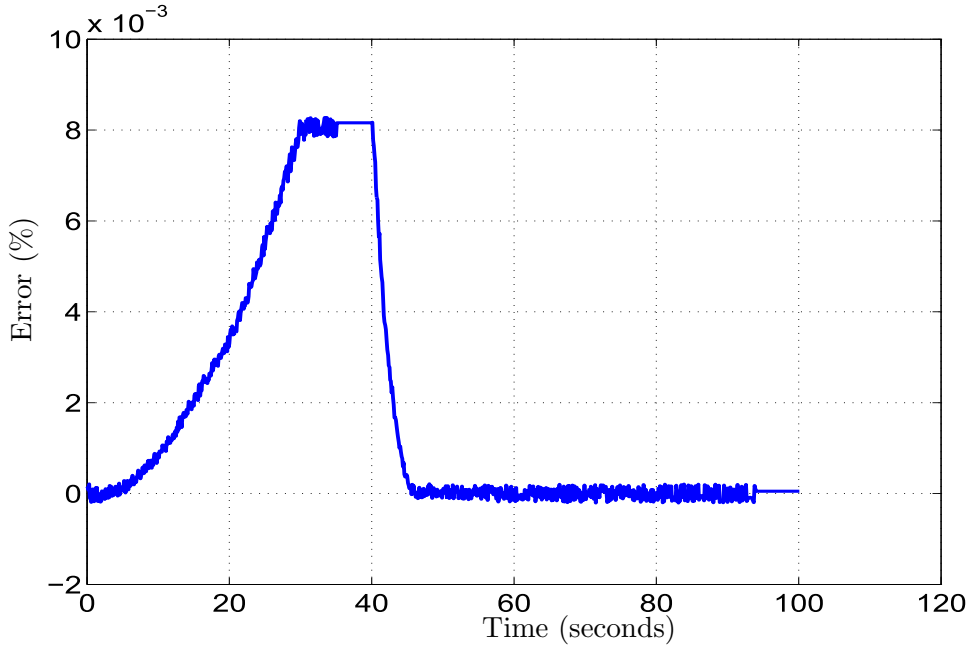
The purpose of this routine is to calculate the total temperature exiting the compressor given inlet total temperature ($T_{t,i}$), pressure ratio (PR) and efficiency (η_c). The routine also serves to return other thermodynamic parameters that are associated with this temperature, namely: specific heats (and ratio), enthalpy, entropy and gas constant. However, these associated parameters are calculated in another routine which has been described and validated in Appendix C. Therefore, this section will only be concerned with the accuracy (compared to GSP) of the exit total temperature calculated.

Transient operation of the engine was used to obtain data from GSP for validation. The values taken from GSP were inlet and exit total temperature as well as pressure ratio and isentropic efficiency. Engine transients were used to subject the routine to the largest possible range and combinations of inputs (i.e. inlet total temperature, pressure ratio and efficiency). The purpose of this was to illuminate any possible vulnerabilities in the routine's calculation. To that end, and for consistency, this validation utilised the extreme transient trajectories (Figures D.19a, D.20a, D.21a and D.22a) used in validating the compressor map handling routines (Appendix D).

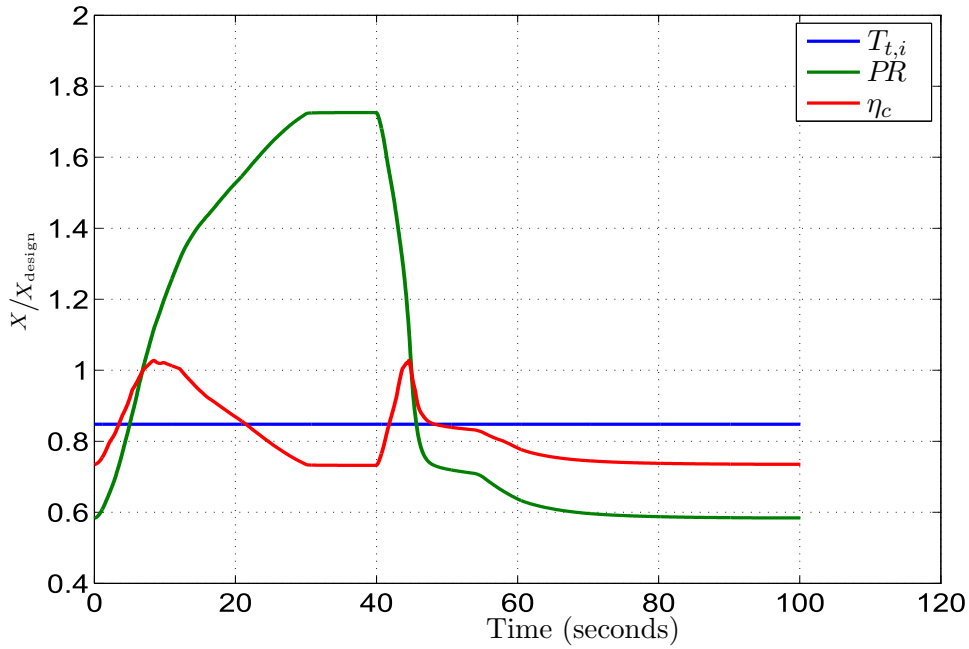
The results of this validation are depicted, for each compressor type, in Figures E.9, E.10, E.11 and E.12. Errors are generally extremely small, as can be seen in Figures E.9a, E.10a and E.11a. However, the error increases dramatically for the high pressure compressor (Figure E.12a in comparison to the others. This increase in error appears to be associated to conditions of high pressure ratio, high inlet temperature and low efficiency, as can be deduced from Figure E.12b. In fact, this trend exists with the other compressor too, i.e. an increase in pressure ratio relates to an increase in error.

It is postulated that there are two possible reasons (or a combination of both) for the discrepancy in results between this routine and that of GSP. The first is that regarding the consideration of variable specific heats. This routine makes the assumption that specific heats are constant across the the compressor, and is calculated iteratively at the average temperature of the compressor. Unfortunately, the approach that GSP uses to deal with variable specific heats has not been included in the user manual or freely available documentation. It is possible that GSP makes use of a different assumption and method for dealing with variable specific heats. The second possibility is that GSP may make use of polytropic efficiency instead of isentropic efficiency, which has also not been documented in the GSP user manual.

Nonetheless, it does not seem necessary to resolve this discrepancy as the error at its maximum does not exceed more than 2%, and is for the most part very accurate. It is on this basis that this routine is deemed valid.

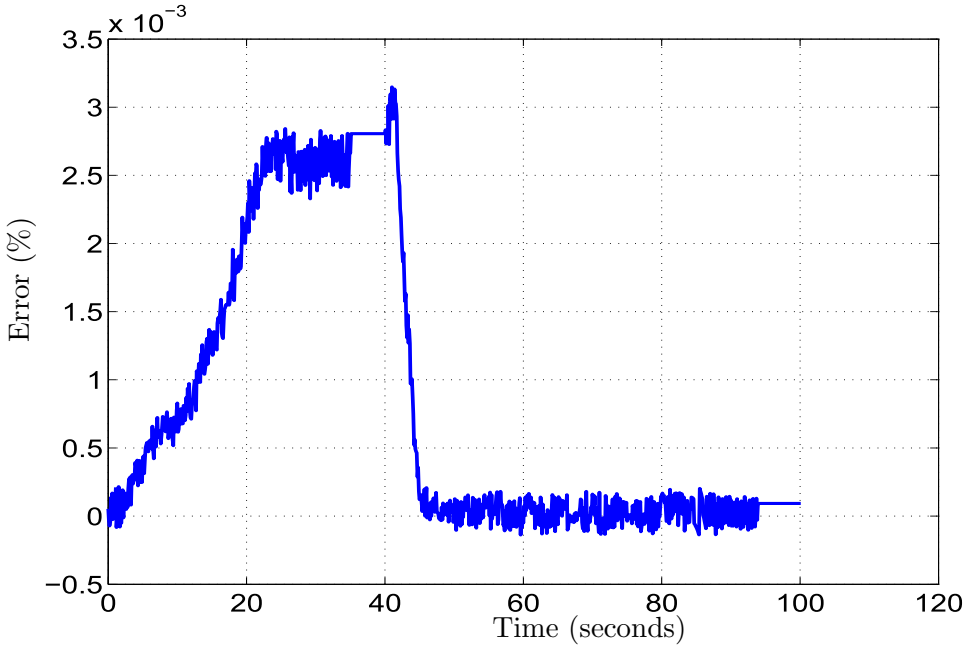


(a) Error of calculated exit temperature

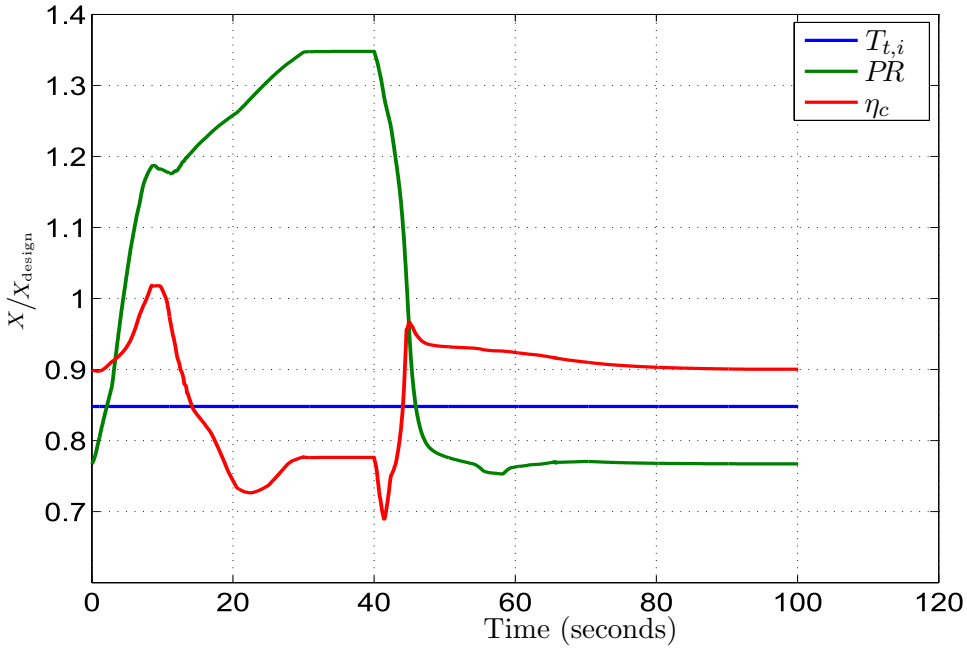


(b) 'CompTherm.m' inputs normalised using design point values, where X represents either $T_{t,i}$, PR or η_c

Figure E.9: Low pressure compressor (bypass) validation of 'CompTherm.m' routine for calculating exit temperature as a function of pressure ratio, efficiency and inlet temperature, corresponding to Figure D.19a

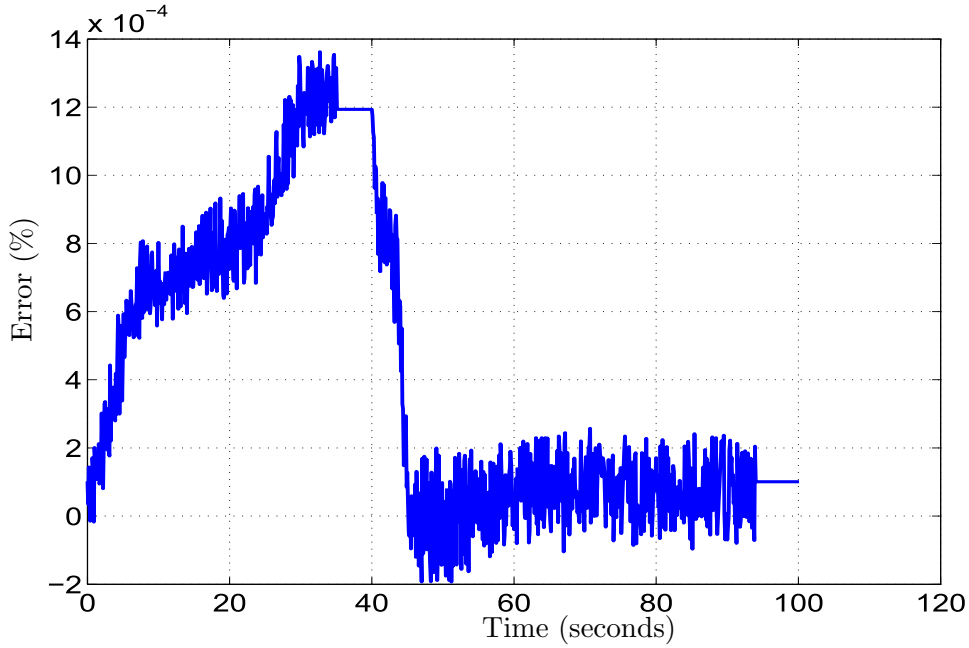


(a) Error of calculated exit temperature

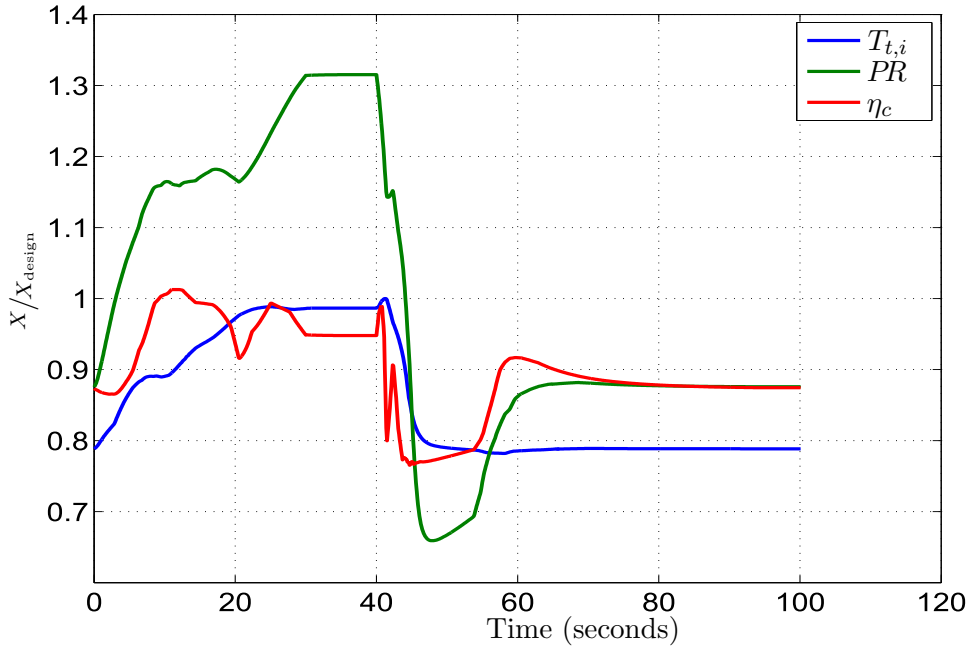


(b) 'CompTherm.m' inputs normalised using design point values, where X represents either $T_{t,i}$, PR or η_c

Figure E.10: Low pressure compressor (core) validation of 'CompTherm.m' routine for calculating exit temperature as a function of pressure ratio, efficiency and inlet temperature, corresponding to Figure D.20a

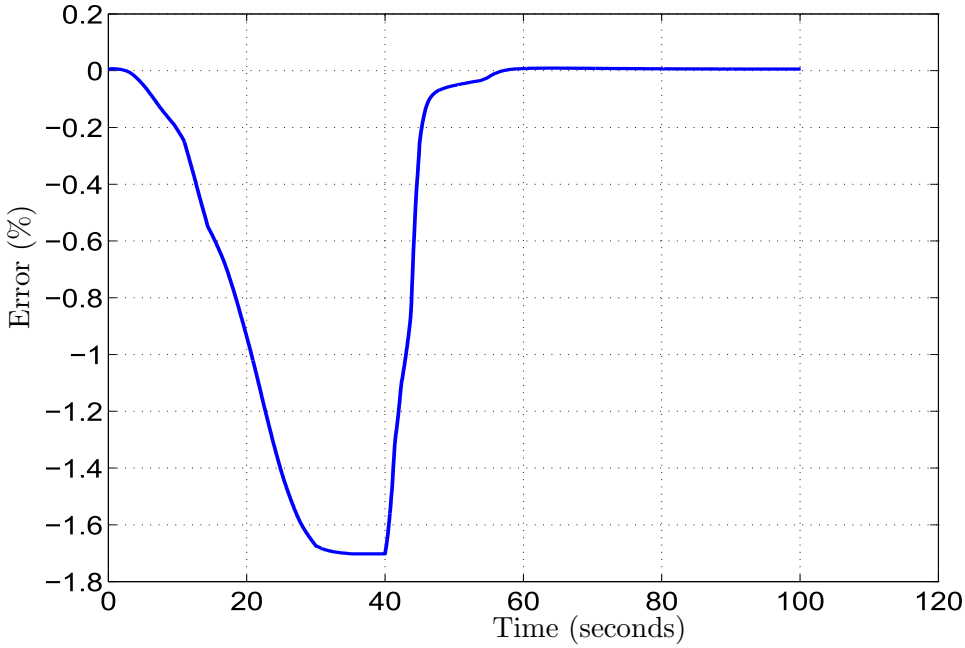


(a) Error of calculated exit temperature

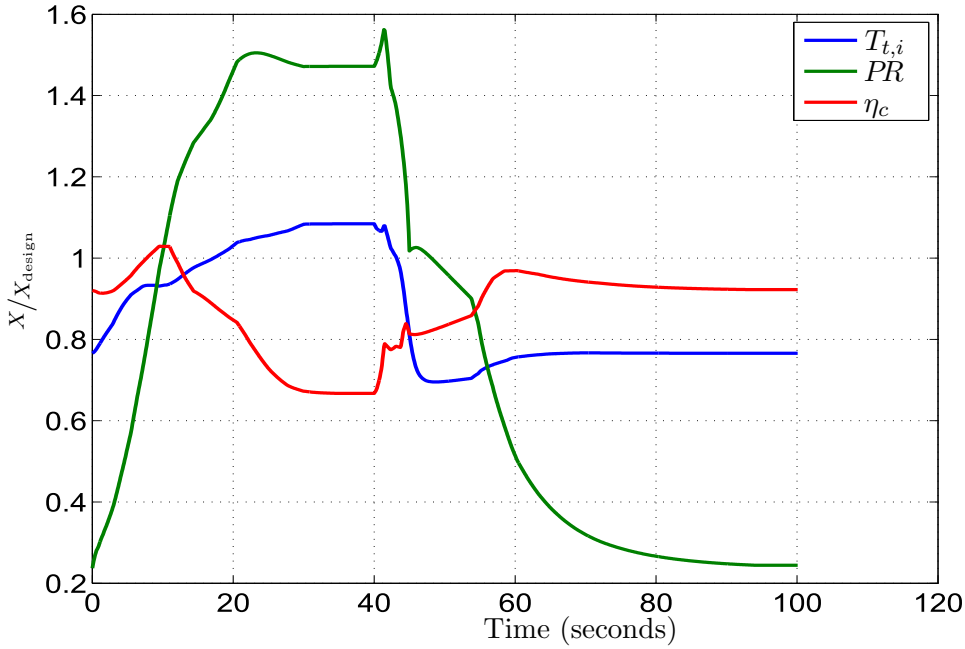


(b) 'CompTherm.m' inputs normalised using design point values, where X represents either $T_{t,i}$, PR or η_c

Figure E.11: Intermediate pressure compressor validation of 'CompTherm.m' routine for calculating exit temperature as a function of pressure ratio, efficiency and inlet temperature, corresponding to Figure D.21a



(a) Error of calculated exit temperature



(b) 'CompTherm.m' inputs normalised using design point values, where X represents either $T_{t,i}$, PR or η_c

Figure E.12: High pressure compressor validation of 'CompTherm.m' routine for calculating exit temperature as a function of pressure ratio, efficiency and inlet temperature, corresponding to Figure D.22a

Compressor.m

At first glance, validation of this routine may seem unnecessary. This is because its purpose is mainly to call the compressor map handling routines (presented in Appendix D) and the routine 'CompTherm.m', both of which have been successfully validated. The only outstanding calculations it performs are those used to determine compressor power E.10 and torque E.11, which are well known and documented. However, it is important to closely examine the results produced by this routine, as errors from its subroutines (which may seem insignificant) may compound upon one another resulting in a significant errors effecting the prediction of engine transient performance.

Therefore, the purpose of this validation is to examine the discrepancies (between this routine and that of GSP) of those parameters that influence the transient operation of the engine. Those parameters are torque (effecting rotor dynamics), mass flow rate (effecting volume dynamics) and exit total temperature (defining the inlet boundary condition of subsequent components). This routine has too been tested according to the transient trajectories of Figures D.19a, D.20a, D.21a and D.22a. This was done so that the routine covers the widest possible range of scenarios that may be encountered by the compressor. It is in addition also consistent with other validations performed thus far, thereby allowing for the source of errors to be tracked more easily and remedied if deemed necessary and possible. The error achieved by this routine (and its subroutines) is depicted in Figures E.13 to E.17.

Figures E.13 and E.14 give errors in calculated mass flow rates and exit temperature, corresponding to the trajectories of Figures D.19a and D.20a, respectively. The results produced by this routine and GSP are very closely correlated, and are well within 1% of one another. The corresponding error of the overall torque of the fan (sum of the torques of the low pressure bypass and core compressors) is depicted in Figure E.15 and is generally found to lie well within 0.5%. The area which produces the highest errors corresponds to a position on the trajectory which occurs outside of the compressor maps bounds (see Figure D.20a).

The routine's accuracy in calculating the operation of the intermediate pressure compressor is depicted in Figure E.16. Errors in mass flow rate and exit temperature are sufficiently small (particularly exit temperature) and fall well within 1%. The torque, on the other hand, depicts some spikes in error which exceed the 1% error achieved by the other parameters. The explanation for this error is that it occurs in a region of the transient trajectory where the pressure ratio dips below unity; this region can be located on Figure D.21a at a corrected speed of between 0.69 an 0.82. When the pressure ratio drops below unity, the compressor effectively acts as a turbine and the temperature ratio also drops between unity, thereby flipping the sign of the associated torque value. In this case the compressor, behaving as a

turbine, has work done on it by the fluid instead of the other way round.

The spikes in torque error are a result of the slightest discrepancy (between GSP and this routine) between the actual points of where the trajectory crosses that line of unity. If, for the same test point, GSP's and this routine's trajectory lie on opposite sides of that line of unity (i.e. one has a pressure ratio above unity and the other below) then the one will yield a positive torque value and the other a negative. This is not reflected in the temperature error, because in either situation (be it larger or lesser than the inlet temperature) its value is always positive and thus the result of both situations will correlate very closely. Nonetheless, because the resultant torques are acting in opposite directions, the error becomes significantly larger. The discrepancy in pressure ratio (which caused this error), is a result of rounding errors when obtaining data from GSP. The pressure ratio in this routine, is calculated from the inlet and exit pressures obtained from GSP, which are rounded to a limited number of significant figures when outputted. Therefore, this error only exists as a result of the validation data, and is not indicative of an actual error in the routine.

Finally, the error achieved in validating this routines implementation on the high pressure compressor, is depicted in Figure E.17. For the most part the errors are generally small, within 1 or 2%. However, there are some rather significan errors at very low and very high speeds. These errors are purely associated to those errors in interpolation of the compressor map, which has been discussed extensively in Section D.1.5. The main cause of these errors is extrapolation outside of the map's bounds at high speeds and pressures, which is evident in Figure D.22a. It has been assumed that the compressor's operation will remain within the maps bounds for this work. The errors which are contained within the maps bounds, have been explained in D.1.5, and is not repeated here. Nonetheless, the average error is small enough.

In summarising, the presence of significantly large errors have been explained and justified. The remaining errors are acceptably small, and therefore the routine is deemed to be valid.

E.4 Fan

Although the fan is essentially a compressor, it needs to be modelled slightly differently. This is because the flow exiting the fan is split between two different flows, meaning that the fan has two different exit boundary conditions. In addition the fan has a much larger diameter than the other compressors, which effects its operation. This section describes how the fan was modelled in this work.

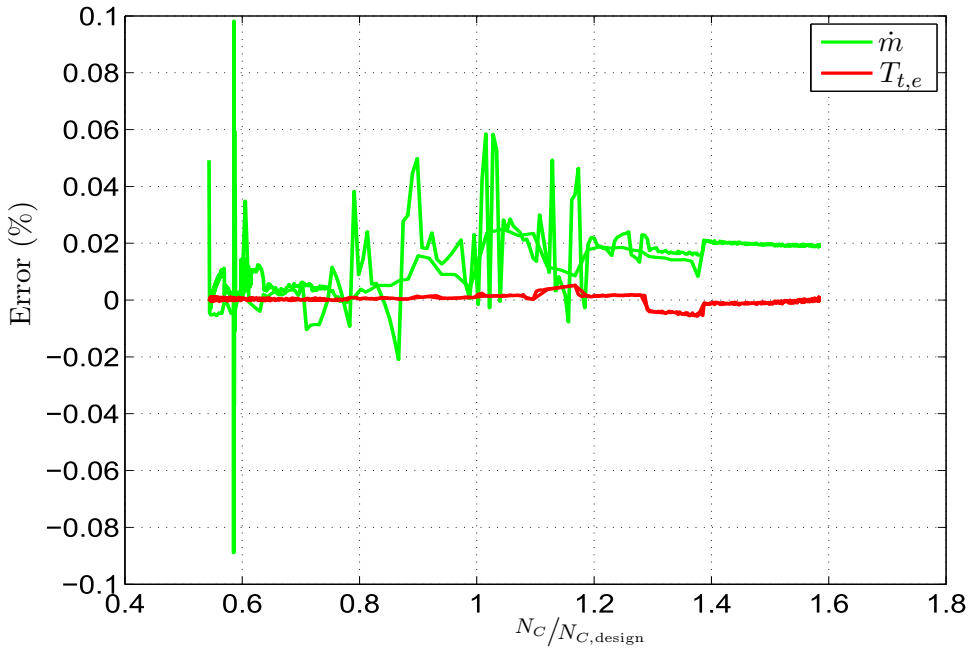


Figure E.13: Low pressure compressor (bypass): Overall compressor routine error compared to GSP

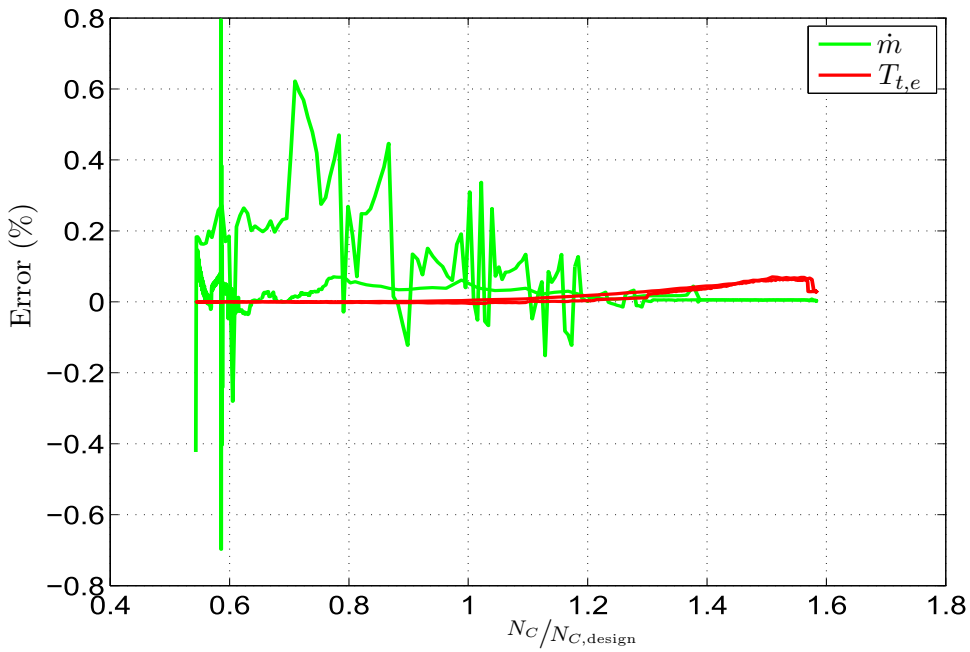


Figure E.14: Low pressure compressor (core): Overall compressor routine error compared to GSP

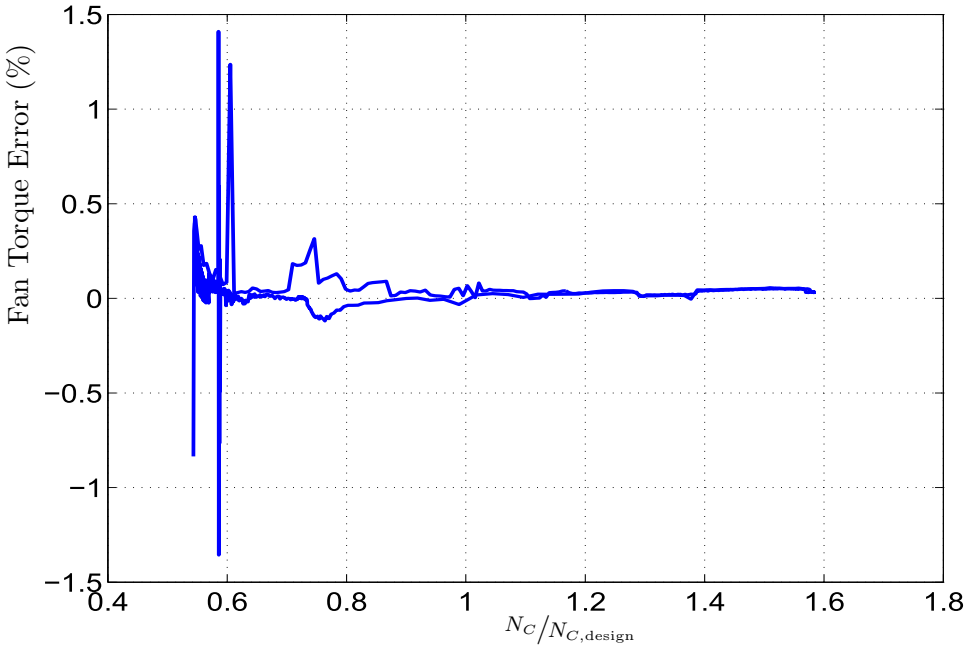


Figure E.15: FAN: Overall compressor routine error (torque) compared to GSP

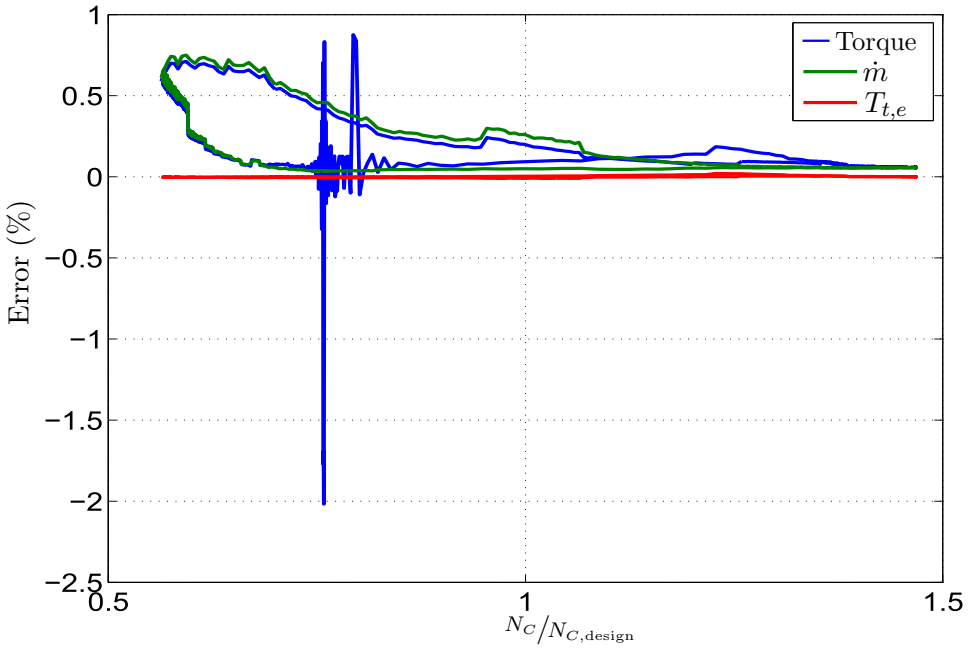


Figure E.16: Intermediate pressure compressor: Overall compressor routine error compared to GSP

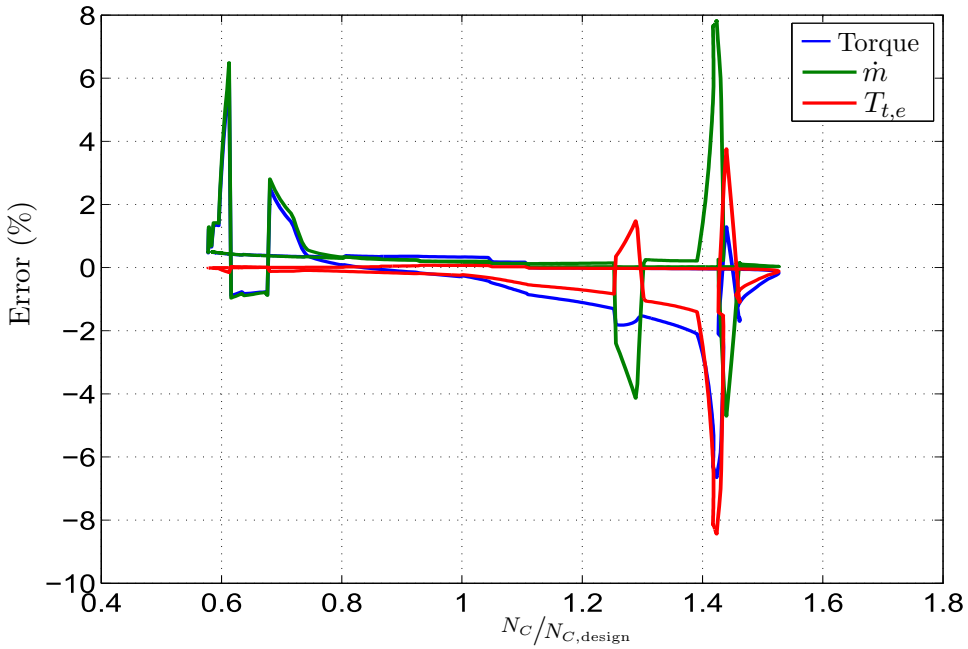


Figure E.17: High pressure compressor: Overall compressor routine error compared to GSP

E.4.1 Theory

With any axial-flow turbo-machine, the flow properties will vary in the axial, radial and circumferential directions. As a simplification, it is generally assumed that axial variations are insignificant and that the radial and circumferential properties may be averaged to yield a reasonable approximation. This assumption forms the basis for use of component performance maps, see Appendix D. However, this approximation is far less accurate when applied to the fan of a high-bypass turbofan engine, due to a number of factors.

Firstly, the fan is the first turbo-machine component to encounter the inlet flow. In reality, the inlet flow is non-uniform, especially in the radial direction. Radial and circumferential non-uniformities are usually due to symmetric and asymmetric boundary layers, respectively. This has already been discussed in Section E.2 on intakes.

Secondly, the diameter of a high-bypass fan is very large and therefore the local blade speed varies significantly from root to tip. This results in highly pronounced radial profiles of flow properties.

Finally, the flow exiting the fan is split between the bypass and core streams. In reality, the average flow properties, between these two streams, differ significantly (due to the fan's pronounced radial profile) and this difference has a large impact on the overall performance of the engine. In

addition, the proportion to which the flow is split between these two flows, varies with off-design (and subsequently transient) engine operation, which too has significant effects on the overall engine performance.

The various methods for overcoming these issues have already been discussed in Section 4.2.3, and will therefore not be repeated here. The method selected for modelling the fan in this work, is one which assumes the fan is effectively comprised of two parallel compressors; one supplying the flow to the core of the engine and the other the to bypass. In so doing, each compressor is described by a different component map, which describes the average flow properties exiting that portion of the compressor. Thus the differences in flow properties entering the core and bypass may be determined more accurately. The discussion on the use of twin compressors, presented here, follows closely the work by Visser [69], which describes GSP's fan model.

The first issue, when using twin maps to represent a compressor, is where to split the fan. This is typically determined according to the engine's design conditions, and in particular the design bypass ratio. The bypass ratio is a ratio of the amount of flow bypassing the core and that passing through the engine core:

$$\mathrm{BPR} = \frac{\dot{m}_{\mathrm{bypass}}}{\dot{m}_{\mathrm{core}}} \tag{E.12}$$

Figure E.18, illustrates how the design bypass ratio splits the fan into two regions, each modelled using a separate performance map (Note that these may be seen as representing physical regions of the map). This representation yields very accurate results when run at design conditions. However, when the engine shifts from its design point, this representation starts to introduce some errors. This is because when the engine deviates from the design conditions, the flows through the bypass and core change, thereby altering the bypass ratio. If the flow entering the fan is assumed to be uniform, then a change in bypass ratio results in a dividing streamline that differs from that of Figure E.18. Therefore, the flow entering the bypass, or core, will pass partially through both compressors, as illustrated by Figure E.19.

Visser [69] has developed a method to somewhat reduce these errors, whilst maintaining the representation of a fan using twin maps. In this method, Visser calculates the amount of flow, entering the bypass duct, that passes through both compressors:

$$\Delta\dot{m}_{bypass} = \dot{m}_i \left[\left(\frac{BPR}{BPR+1} \right)_{\text{design}} - \left(\frac{BPR}{BPR+1} \right)_{\text{off-design}} \right] \tag{E.13}$$

Equation E.13, is then used to determine the effective flow through each of the fan's two maps. These are given by Equations E.14 and E.15, where

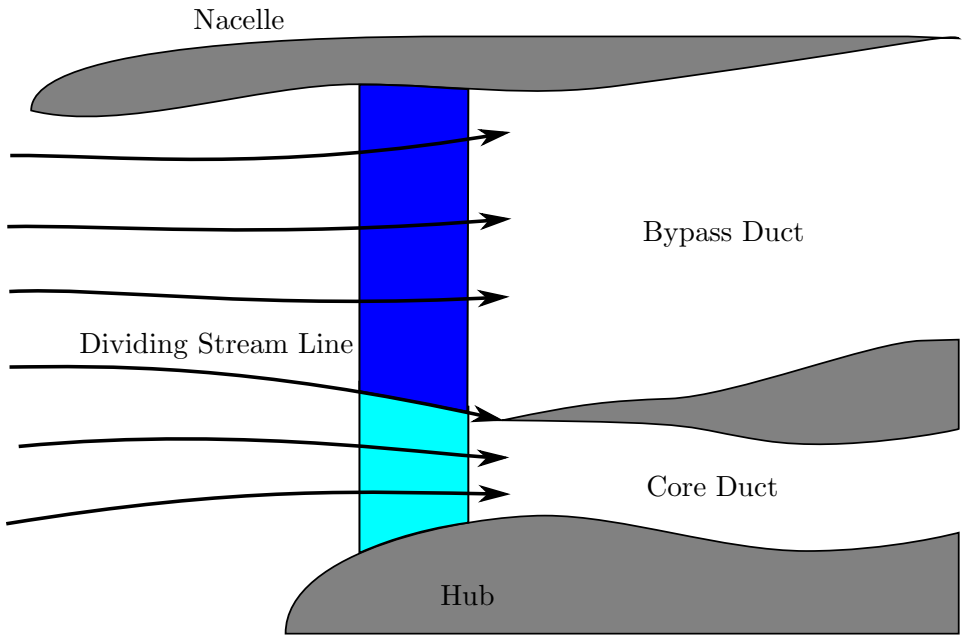


Figure E.18: Fan: Parralel compressors split at design BPR, with dark and light blue representing the physical portions of the fan that are modelled by different performance maps

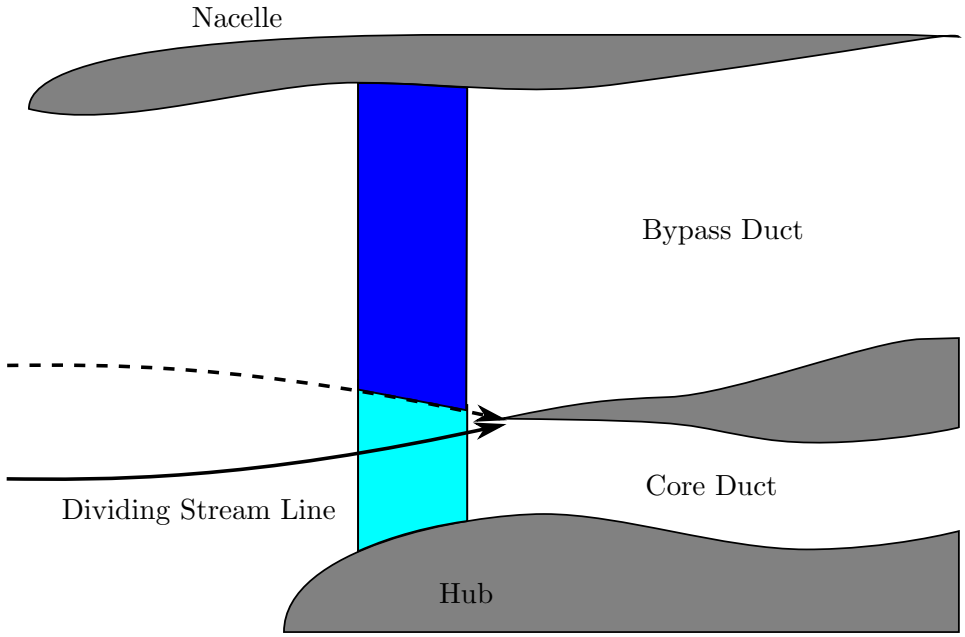


Figure E.19: Fan: Parralel compressors split at off-design BPR, with a shift in the position of the dividing streamline

a correction factor (cf) is a user-defined constant that specifies the contribution of each map to the compression of $\Delta\dot{m}_{bypass}$.

$$\dot{m}_{\text{bypass, effective}} = \dot{m}_i \left(\frac{BPR}{BPR+1}\right)_{\text{design}} + (\text{cf} \times \Delta\dot{m}_{bypass}) \tag{E.14}$$

$$\dot{m}_{\text{core, effective}} = \dot{m}_i \left(\frac{1}{BPR+1}\right)_{\text{design}} - (\text{cf} \times \Delta\dot{m}_{bypass}) \tag{E.15}$$

The correction factor can be a constant anywhere between 0 and 1. If cf = 0, there is no cross-flow between compressors, and the flow is assumed to adjust itself after exiting the fan and before splitter. On the other extreme, if cf = 1, there is also no cross-flow between compressors, and the flow is assumed to adjust itself before entering the fan. In reality, cf will exist somewhere between these two values, and represents the distance between the fan's exit and the flow split. Therefore, cf needs to be determined via trial and error for a given engine.

However, this method does not integrate well with the Inter-Component Volume Method of modelling engine transients. This is because Visser's method defines compressor operation as a function of spool speed and mass flow rate, which is then used to determine the pressure ratio across the compressor. Whereas, the Inter-Component Volume Method defines compressor performance as a function of spool speed and pressure ratio, which is used to determine the mass flow rate through the compressor. Therefore, the fan has been modelled assuming a correction factor of unity, which allows the Inter-Component Volume Method to determine the bypass ratio as follows:

$$\text{BPR} = \frac{\dot{m}_{\text{bypass}}}{\dot{m}_{\text{core}}} = \frac{f\langle N,\ PR_{\text{bypass}}\rangle}{f\langle N,\ PR_{\text{core}}\rangle} \tag{E.16}$$

Therefore, it is assumed that the fan operates as though there were a shroud separating the two portions of the fan, thereby forcing non-uniformities in the intake to accommodate variations in off-design bypass ratio. This is depicted in Figure E.20.

E.4.2 Simulink Model

The Simulink model, developed to simulate the fan, is simply an assembly of compressor components, and has therefore not been included here, as it is dependent on the assembly of the entire engine model.

E.4.3 MATLAB Code

There is no additional code to that contained within the compressor components, of which the fan is comprised.

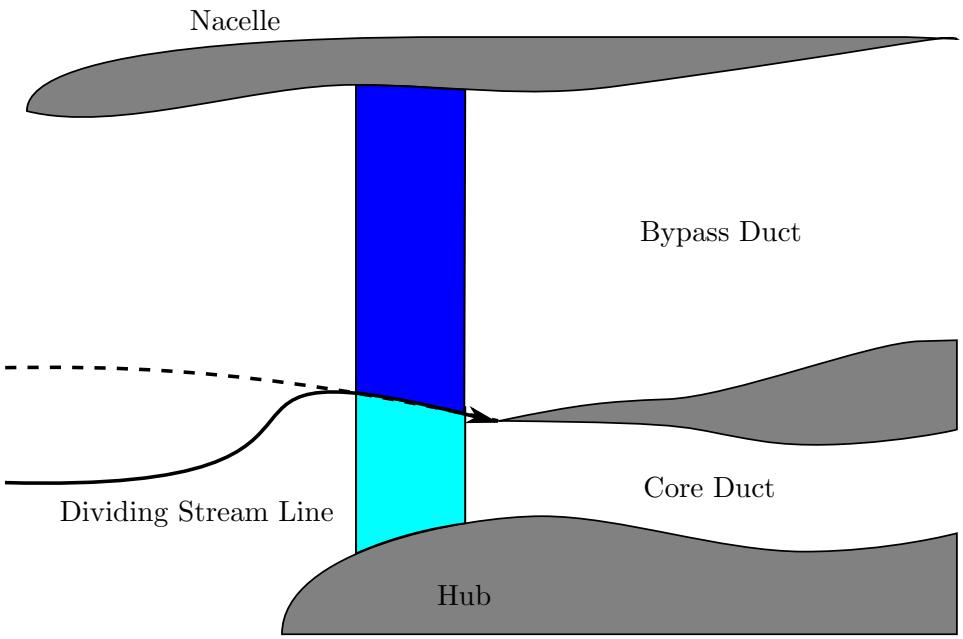


Figure E.20: Fan: Parralel compressors assumption: Assumes flow adjusts within the intake to prevent cross-flow between compressors.

E.4.4 Validation

The fan is comprised of two compressor model's, which have already been validated. The only parameter which has not been validated is the bypass ratio. However this is dependent on the operation of the entire engine mode, and therefore its validation is incorporated with that of the entire engine model. Furthermore, when validating the engine model against that of GSP, a correction factor of unity was applied to the GSP model too.

E.5 Combustion

E.5.1 Theory

The combustion process is very complex, and involves a combination of time dependent chemical reactions and fluid dynamics. Therefore, very sophisticated models may be developed to simulate mixing of fuel, air and combustion products and the corresponding chemical reactions. Such a model requires extensive knowledge on the geometry and boundary conditions of the combustion chambers to conduct high fidelity CFD simulations of the mixing process. Simultaneously, the chemical reactions (and subsequent thermodynamic changes) of each fluid element need to be accounted for, according to their mixture properties. This will allow for accurate predictions

of the gas composition and thermodynamic properties exiting the combustion chambers, which may then be used in determining the operation of the engine's turbines.

However, developing such a model requires a huge amount of time, effort and expertise. In addition, should such a model be available, it is bound to be highly computationally demanding and therefore impractical for inclusion within a transient engine model [4]. In addition, such a high-fidelity combustion model requires high-fidelity models of its adjacent components, such that its boundary conditions are correctly defined. This, therefore, results in the requirement of a hugely complex engine model, which is not practical for the purposes of this work.

Instead, it is convenient to make some simple assumptions about the combustion process. These assumptions allow for the geometry of the combustion chamber/s (and therefore mixing) to be ignored, as well as simplifying the description of the chemical reaction taking place.

It is assumed that air and fuel enter the combustion chamber as a perfectly uniform mixture (i.e. air and fuel mix ideally and instantaneously), according to air and fuel flow rates at an instance in time. Furthermore, it is assumed that combustion of the reactants occurs instantaneously, and that no mixing occurs between reactants and products. In other words, for an instance in time, the reactant mixture inside the combustion chamber is assumed to be purely a function of the incoming air-fuel ratio, and ignores any combustion products that still may exist from the previous instance in time. In addition, the combustion products are assumed to be time invariant (i.e. the combustion products are always assumed to be in chemical equilibrium), and therefore dissociation effects may be ignored. And finally, it is assumed that the combustion products exit the combustion chamber as a perfectly uniform mixture. In effect, these assumption result in a steady-state combustion model, ignoring mixing and dissociation effects.

Based on the above assumptions, the properties of the gas mixture exiting the combustion chamber/s can be easily calculated. In this calculation, the aim is to determine the exiting gas' composition and total temperature and pressure as a function of the inlet conditions (air-fuel ratio and total temperature and pressure). These properties are necessary for calculating the operation of downstream components.

Combustion in a gas turbine engine, typically, occurs at constant pressure (Justifying neglecting effects of dissociation [130]). Of course in reality there will no doubt be pressure losses due to turbulent mixing for example. Nonetheless, this can be crudely accounted for by including a pressure ratio term when calculating the exit pressure:

[4] This would require either (or a combination thereof) parallel or distributed computing as well as a specifically designed programming architecture and interface. Some architectures do exist too aid this type of multi-fidelity simulation, examples of which are PROOSIS [77, 107, 108] and NPSS [84, 85, 97, 98]

$$P_{t,e} = PR_{\,b} P_{t,i} \tag{E.17}$$

This pressure ratio may be found from engine test data for different steady-state operating conditions, and used to form look-up tables which may in-turn be applied to a transient engine model. Such a look-up table may express combustion pressure losses as a function of air-fuel ratio and inlet mass flow rate from the high pressure compressor. However, it is assumed that pressure losses in the combustion chamber will not vary drastically across the operating envelope of the engine, and therefore may be held constant at its design value.

The remaining outputs, total temperature and exit gas composition, is then determined from combustion equations. If combustion is ideal, it means that all the fuel is burnt (no excess fuel left over), reacting with all of the oxygen (no excess oxygen left over). This type of combustion is referred to as stoichiometric combustion:

$$C_x H_y + a\,(O_2 + 3.76 N_2) = xCO_2 + \frac{y}{2} H_2O + 3.76 a N_2 \tag{E.18}$$

where $a = x + {}^{y}/_{4}$. The fuel-air ratio, required to meet stoichiometric conditions is given as a ratio of fuel-air molecular weights, corresponding to Equation E.18,

$$\left(\frac{F}{A}\right)_{\text{stoich}} = \frac{\text{MW}_{\text{fuel}}}{a(1+3.76)\text{MW}_{\text{air}}} \tag{E.19}$$

However, stoichiometric combustion is unlikely and combustion will typically occur under either fuel-rich or fuel-lean conditions. The equivalence ratio is is used to compare the actual fuel-air ratio against that of stoichiometric conditions [5]:

$$\phi = \frac{\left(\frac{F}{A}\right)_{\text{actual}}}{\left(\frac{F}{A}\right)_{\text{stoich}}} \tag{E.20}$$

where $\phi > 1$ and $\phi < 1$ indicates a fuel-rich and fuel-lean mixture, respectively. The combustion equation, Equation E.18, must be adjusted for equilibrium to account for excess fuel ($\phi > 1$) or oxygen ($\phi < 1$), as indicated by Equations E.21 and E.22, respectively.

$$C_x H_y + \frac{a}{\phi}\,(O_2 + 3.76 N_2) = \frac{x}{\phi} CO_2 + \frac{y}{2\phi} H_2O + \frac{3.76a}{\phi} N_2 + \frac{a}{(1-\phi^{-1})} C_x H_y \tag{E.21}$$

[5] The inverse of the equivalence ratio, commonly denoted as λ, is often used instead for convenience, but this work will only make use of ϕ

$$C_xH_y + \frac{a}{\phi}(O_2 + 3.76N_2) = xCO_2 + \frac{y}{2}H_2O + 3.76aN_2 + \frac{a}{(\phi^{-1} - 1)}O_2 \quad \text{(E.22)}$$

Therefore, the composition of the combustion products is determined from either Equation E.18, E.21 or E.22. The temperature of the combined combustion products may then be calculated according to this composition based on what is referred to as the adiabatic flame temperature. The adiabatic flame temperature, at constant pressure, gives the temperature of the combustion products assuming that there has been no transfer of heat, kinetic or potential energy to or from the system. The temperature is calculated by applying these assumptions to the first law of thermodynamics for a closed system:

$$\Delta(qm) - \Delta W = \Delta E \quad \text{(E.23)}$$

It is assumed that pressure is constant throughout the combustion process, and therefore (because it is assumed that there is no change in kinetic or potential energy) the change in work in the system is due to the change in volume between the reactants and products:

$$\Delta W = P \int_{\text{reac}}^{\text{prod}} \mathrm{d}V = P\,(V_{\text{prod}} - V_{\text{reac}}) \quad \text{(E.24)}$$

Finally, because an adiabatic process is assumed ($\Delta qm = 0$) and that mass is conserved, Equation E.23 can be simplified to yield Equation E.25.

$$h_{\text{prod}} = h_{\text{reac}} \quad \text{(E.25)}$$

Equation E.25, does not hold true if the products and reactants are at the same temperature, remembering that enthalpy is indeed a function of temperature. This is because the molecules from which the products and reactants are comprised, all have different enthalpies of formation (i.e. some require a larger amount of energy to form, and remain in a state of equilibrium, than others). Therefore, knowing the temperature of the reactants, there exists a temperature at which the products must exist for Equation E.25 to hold true. This temperature is the Adiabatic Flame Temperature at constant pressure (AFT_P), and may be used to represent the temperature of the gas mixture exiting the combustion chamber/s.

Because enthalpy is a function of temperature, the AFT_P must be calculated iteratively. A simple bisection method has been utilized to perform this iterative calculation, because of its speed to convergence and robustness. The interval wherein a solution exist is well known, therefore allowing for confident usage of the bisection method. The intervals lower boundary is chosen to be equal to the temperature of the inlet air to the combustion

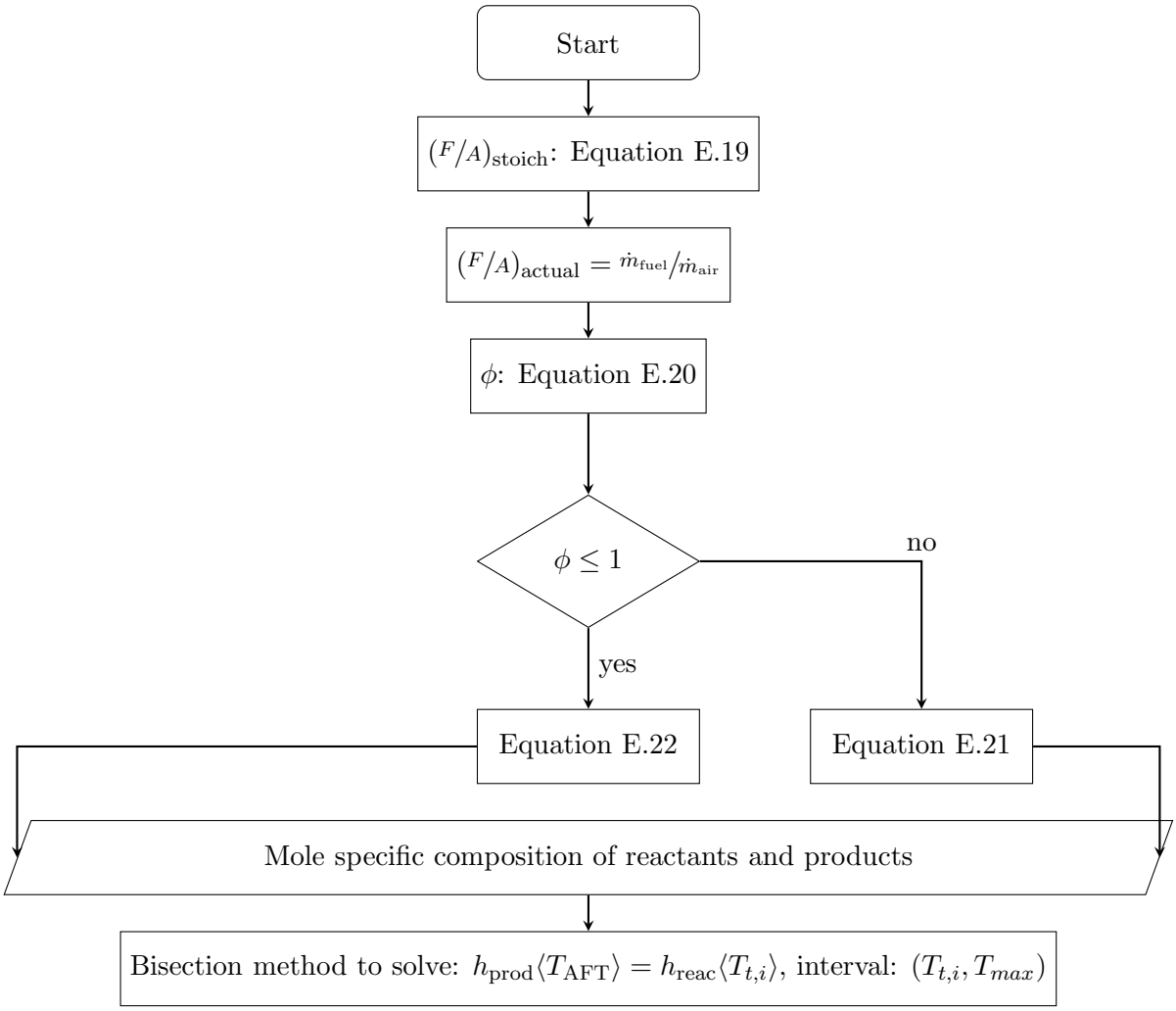


Figure E.21: Flow chart: Adiabatic flame temperature calculation

chamber/s, as it is assumed that the adiabatic flame temperature will always be greater than this value. The upper bound of this interval is given as the highest temperature which is allowed by the gas property tables. It is highly unlikely, and probably impossible, that the engine could be operated such that it exceeds theses bounds.

E.5.2 Simulink Model

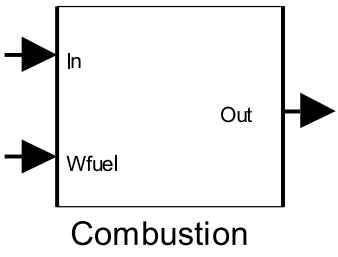
The Simulink model developed to represent the combustion process, is depicted in Figure E.22. Figure E.22a, depicts the highest level of the block, which is used for interfacing with other component models. This is usually the inter-component volume block or a compressor block - depending on the

intended modelling technique of the user and the configuration of the engine being modelled - and the turbine block. This high-level block takes in the fuel flow rate ($^{kg}/_{s}$), in the first inlet port, and the exit conditions from an upstream component into its second inlet port.

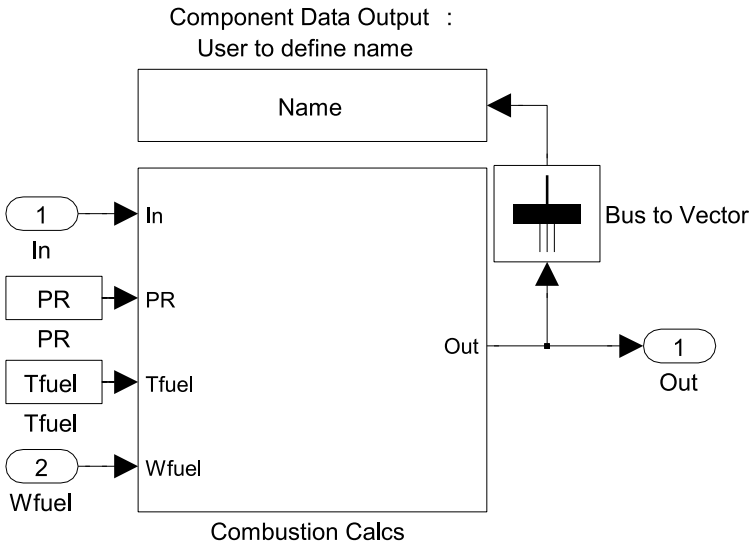
The subsystem in Figure E.22a is depicted in Figure E.22b, and represents the user interface level. In this subsystem, the user is able to define the pressure ratio (which is held constant) at which the combustion process occurs, as well as the temperature at which the fuel is injected (also held constant). Future users may wish to change the pressure ratio and fuel temperature dynamically during the simulations, and this will require some modification of the this subsystem. The modifications to allow for dynamic pressure ratio are likely to be a bit complex, and is highly dependent on the method of predicting changes in pressure ratio. The temperature ratio of the fuel is a bit simpler, if one considers how the fuel's temperature may change in reality. Typically, a simple assumption would be to assume that the fuel temperature will be the same as that of the local ambient temperature, and therefore only change with altitude. This has been ignored in this particular work, because the aircraft are assumed to fly at a fixed altitude. Nevertheless, one may simply include an additional inlet port to the subsystem which takes in the ambient temperature calculated in the Flight Conditions Block (Figure E.2).

Figure E.22c, depicts the 'Combustion Calcs' subsystem of Figure E.22b. This subsystem extracts the exit pressure, temperature and mass flow rate parameters from the bus signal passed to the combustion block from an upstream component. These parameters, along with the specified pressure ratio, fuel temperature and fuel flow rate, are past to an Embedded MATLAB Function ('Combustion AFT Calcs'), which is used as an interface between the Simulink and MATLAB environments. Therefore, the Embedded MATLAB Function is used to pass inputs to MATLAB function files for calculating the outputs of the combustion process. This Embedded MATLAB Function, then returns the calculated outputs to the Simulink environment. These outputs describe the combustion products and include total exit pressure and temperature, mass flow rate, the thermodynamic properties (specific heats, enthalpy, entropy and gas constant) and the composition (A vector of Mole fractions denoted 'GasMix'). These outputs are grouped into a 'bus' signal which is passed to out of the subsystem and the entire combustion block to be passed on to downstream components. This bus signal, as depicted in Figure E.22b, is also converted to a structure and saved to the MATLAB Workspace (under a user-defined name), thereby recording the changes in output parameters as the simulation progresses.

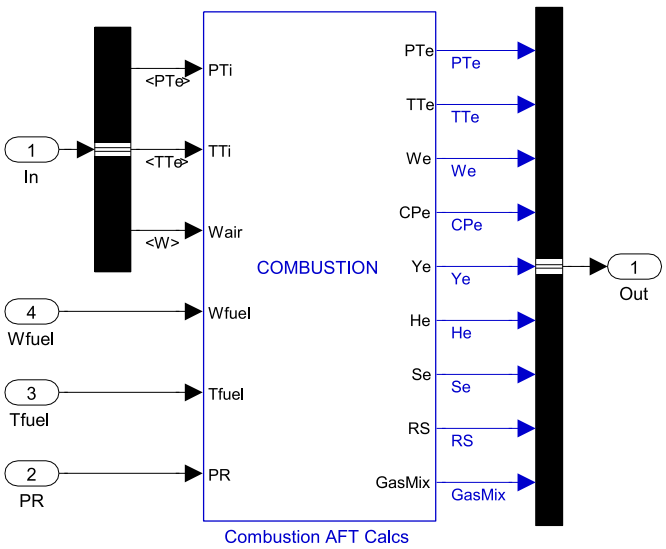
The Embedded MATLAB Function, and other MATALB functions which perform the actual combustion calculations, are detailed in the section that follows.



(a) High-level combustion block



(b) User-interface level



(c) Combustion calculations subsystem

Figure E.22: Combustion model blocks developed in Simulink

E.5.3 MATLAB Code

Code Listing E.6 details the contents of the Embedded MATLAB Function depicted in Figure E.22c. This function performs two very simple calculations itself; Determines the exit pressure, according to the inlet pressure and pressure ratio, and exit mass flow rate, which is a sum of the inlet flow rates of air and fuel. This function is then used to call a function 'CombChem.m', which is used to calculate the properties of the combustion products.

Code Listing E.6: Embedded MATLAB Function COMBUSTION

```
%=================================================================
%                          COMBUSTION BLOCK
%=================================================================
%
% NAME:         COMBUSTION
%
% AUTHOR:       D. Sanders, University of Cape Town
%
% DATE:         21 November 2012
%
% FUNCTION:     [TTe, PTe, GasMix, We, Rs] = COMBUSTION ...
    (PTi, TTi, Wair, Wfuel, Tfuel, PR)
%
% PURPOSE:      Call combustion calculation routine from ...
    MATLAB and calculate exit mass flow rate and total ...
    pressure.
%
% INPUTS:
%
%   PTi         = Inlet Pressure                    [Pa]
%   TTi         = Inlet Temperature                 [K]
%   Wair        = Inlet Air Mass Flow Rate          [kg/s]
%   Wfuel       = Inlet Fuel Mass Flow Rate         [kg/s]
%   Tfuel       = Inlet Fuel Temperature            [K]
%   PR          = Pressure Loss Ratio               [-]
%
% OUTPUTS:
%
%   TTe         = Exit Temperature                  [K]
%   PTe         = Exit Pressure                     [Pa]
%   GasMix      = Vector of Gas Composition [Mol Fractions]
%   We          = Exit Mass Flow Rate               [kg/s]
%   Rs          = Specific Gas Constant             [J/kg-K]
%
% SUBROUTINES:  CombChem.m
%
% REFERENCES:
%
% For coding conventions, please refer to:
%
%   Johnson, Richard. "MATLAB Programming Style Guidlines".
```

```
39 %   ver 1.5. October 2002.
40 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
41 %=============================================================
42
43 function [PTe, TTe, We, CPe, Ye, He, Se, RS, GasMix] = ...
       COMBUSTION (PTi, TTi, Wair, Wfuel, Tfuel, PR)
44
45 %-------------------------------------------------------------
46 %               Block initialisation & Calculations
47 %-------------------------------------------------------------
48
49 % MATLAB function to be called
50 eml.extrinsic('CombChem');
51 TTe = 0; CPe = 0; Ye = 0; He = 0; Se = 0; RS = 0; GasMix = ...
       ones(5,1); PTe = 0; We = 0;
52
53 % Calculate Exit Pressure
54 PTe = PR * PTi;
55
56 % Calculate Exit Flow Rate
57 We = Wair + Wfuel;
58
59 % Calculate Combustion Exit Conditions
60 [TTe, CPe, Ye, He, Se, RS, GasMix] = CombChem (TTi, Tfuel, ...
       Wair, Wfuel);
```

'CombChem.m' is presented in Code Listing E.7, and is used in performing the combustion calculations outlined in the theory section above, and depicted in the flow chart of Figure E.21. This function determines whether the combustion is fuel-rich or fuel-lean, thereby calculating the Mole fraction composition of the combustion products. The bisection method is then applied, based on the calculated composition, to determine the temperature of the combustion products. In so doing, this function calls a subroutine 'CombTherm.m' [6], which is used to determine the enthalpy of the combustion products according to the calculated composition and estimated temperature.

Code Listing E.7: MATLAB Function CombChem.m: For performing combustion calculations

```
1 %=============================================================
2 %                      COMBUSTION CHEMISTRY
3 %=============================================================
4 %
5 % NAME:          CombChem.m
6 %
7 % AUTHOR:        D. Sanders, University of Cape Town
8 %
9 % DATE:          15 October 2012
```

[6] 'CombThermm.m' has already been discussed in detail in Appendix C

```
10 %
11 % FUNCTION:     [TTe, CPe, Ye, He, Se, NP] = CombChem ...
       (Tair, Tfuel, wair, wfuel)
12 %
13 % PURPOSE:      Calculates the temperature and composition ...
       of combustion products for varying fuel-air-ratiois. ...
       Calculations are based on the Adiabatic flame ...
       temperature method.
14 %
15 % INPUTS:
16 %=
17 %   Tair        = Inlet air temperature             [K]
18 %   Tfuel       = Inlet fuel temperature            [K]
19 %   wair        = Inlet mass flow rate of air       [kg/s]
20 %   wfuel       = Inlet mass flow rate of fuel      [kg/s]
21 %
22 % OUTPUTS:
23 %
24 %   TTe         = Exit Temperature                  [K]
25 %   TUA         = Temperature ratio                 [-]
26 %   CPe         = Exit specific heats               [kJ/kg-K]
27 %   Ye          = Exit specific heats ratio         [-]
28 %   He          = Exit Enthalpy                     [kJ/kg]
29 %   Se          = Exit Entropy                      [-]
30 %   NP          = Mole fraction composition of combustion ...
       products: Vector [FUEL O2 N2 CO2 H2O]
31 %
32 % SUBROUTINES:  CombTherm.m
33 %
34 % REFERENCES:   Calculations based on:
35 %
36 %               Turns, S.
37 %               "An Introduction to Combustion: Concepts ...
       and Applications" McGraw-hill New York, 1996
38 %
39 % For coding conventions, please refer to:
40 %
41 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
42 %   ver 1.5. October 2002.
43 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
44 %===============================================================
45 function [TTe, CPe, Ye, He, Se, RS, NP] = ...
46             CombChem (Tair, Tfuel, wair, wfuel)
47 %===============================================================
48
49 %---------------------------------------------------------------
50 %                   Fuel Properties: Jet-A
51 %---------------------------------------------------------------
52     % Molecular weight                              [kg/kmol]
53     MWfuel = 167.3110200;
54
55     % Chemical Formula
56     % CxHy + a(O2 + 3.76N2) --> xCO2 + (y/2)H2O + 3.76aN2
57     x = 12; y = 23; a = x + y/4;
```

```
58
59 %-------------------------------------------------------------------------
60 %                          Air Properties
61 %-------------------------------------------------------------------------
62     % Molecular weight                                [kg/kmol]
63     MWair = 28.96512;   % Actual
64
65     % Molar Ratio of N2 to O2                         [-]
66     z = 3.773;
67
68 %-------------------------------------------------------------------------
69 %                      Mixture Properties
70 %-------------------------------------------------------------------------
71     % Fuel air ratios for mix and at stoichiometry
72     FAR = wfuel/wair;
73     FARs = MWfuel / ((1+z) * a * MWair);
74
75     % Equivelence ratio
76     phi = FAR / FARs;
77
78     % Select between Stoichiometric, Lean or Rich mixtures
79     % Stoichiometric or Lean
80     if phi <= 1;
81
82         % Specific Moles of Reactants and Products
83         % [FUEL O2 N2 CO2 H2O]
84         NR = [1; a/phi; z*a/phi; 0; 0];
85         NP = [0; a*(1/phi - 1); z*a/phi; x; y/2];
86
87     % Rich
88     elseif phi > 1
89
90         % Specific Moles of Reactants and Products
91         % [FUEL O2 N2 CO2 H2O]
92         NR = [1; a/phi; z*a/phi; 0; 0];
93         NP = [(1 - 1/phi); 0; z*a/phi; x/phi; y/2/phi];
94
95     end
96
97 %-------------------------------------------------------------------------
98 %           AFT Calculation - Bisection Method
99 %-------------------------------------------------------------------------
100
101    % Enthalpy of Reactants
102    [CPR, YR, HR, SR, RS] = CombTherm (Tfuel, Tair, NR);
103
104    % Enthalpy of products
105    % Initialise bisection method interval
106    x1 = Tair; % inlet temp of combustor
107    x2 = 6000; % Maximum allowable temperature
108
109    % Enthalpy [kJ]
110    [CP1, Y1, H1, S1, RS] = CombTherm (x1, x1, NP);
111    [CP2, Y2, H2, S2, RS] = CombTherm (x2, x2, NP);
```

```
112
113     deltH1 = (H1 - HR);
114     deltH2 = (H2 - HR);
115
116     if H1 == H2
117         warning('Temperature interval boundaries give same ...
                enthalpy')
118         return
119     elseif deltH1 == 0
120         warning('Lower temperature interval is T_AFT')
121         T_AFT = x1;
122         return
123     elseif deltH2 == 0
124         warning('Upper temperature interval is T_AFT')
125         T_AFT = x2;
126         return
127     elseif deltH1*deltH2 > 0
128         warning('No solution within specified temperature ...
                interval, or interval is too large')
129         return
130     end
131
132     T_AFT_tol = 0.1; % K
133     iter = 0;
134
135     while abs(x1-x2) > T_AFT_tol
136
137         xm = (x1 + x2)/2;
138
139         [CP1, Y1, H1, S1, RS] = CombTherm (x1, x1, NP);
140         [CP2, Y2, H2, S2, RS] = CombTherm (x2, x2, NP);
141         [CPm, Ym, Hm, Sm, RS] = CombTherm (xm, xm, NP);
142
143         deltH1 = (H1 - HR);
144         deltH2 = (H2 - HR);
145         deltHm = (Hm - HR);
146
147         T_AFT = xm;
148
149         if deltHm*deltH1 < 0
150
151             x2 = xm;
152
153         elseif deltHm*deltH2 < 0
154
155             x1 = xm;
156
157         else
158             error('AFT calc Failed to converge to T_AFT')
159             return
160         end
161
162         iter = iter + 1;
163
```

```
164     end
165
166     TTe = T_AFT; CPe = CPm; Ye = Ym; He = Hm; Se = Sm;
167
168     return
```

E.5.4 Validation

The combustion model developed in this section was compared to that of GSP. Before looking at the results of the validation tests, it is important to be aware of the level of sophistication of the GSP model to which this model is compared.

Firstly, the number of chemical species contained in its library is far greater than the library developed for this work (see Appendix C). For example, this work ignores the presence of argon in air (as well as other molecules such as CO_2), and instead makes use of atmospheric/apparent nitrogen, who's volume accounts for the inert gases found in dry air. This is a perfectly acceptable approximation, which is used in well respected and popular texts such as References [68] and [130]. GSP on the other hand, has included a more accurate composition of air, therefore taking into account almost all of the constituents of dry air.

Secondly, GSP incorporates the effects of dissociation into its combustion calculation. Combustion products change composition during the combustion process. Effectively, the combustion products react with one another. As an example, CO_2 dissociates into CO and O_2 in a reversible process:

$$CO_2 \rightleftharpoons CO + \frac{1}{2}O_2 \tag{E.26}$$

During combustion this reaction is operating in both directions in an attempt to reach a state of equilibrium. However this takes time, and in-fact the different reaction directions occur at different rates. It may be assumed that combustion occurs so rapidly that the products of Equations E.18, E.22 or E.21, do not have enough time to dissociate, and the products remain the same. This is the assumption made in the model developed in this section. Alternatively, it may be assumed that combustion occurs slowly enough to allow for the combustion products to completely react with one another until equilibrium is reached (dissociation occurs completely). In this case it is important to note that additional species have been formed within the combustion products, and that their formation enthalpy is inherently different. Typically, the enthalpy of formation of the dissociated species will be higher than that of the original species, and therefore heat has been 'absorbed' in forming them. This means that the adiabatic flame temperature of a dissociated mixture is lower than that of mixture that has not had any time to dissociate. In reality, however, combustion may occur at a rate between

these two extremes. This results in partial dissociation and, because the opposite directions of reactions occur at different rates, a mixture that is not in equilibrium. It appears that GSP allows for the complete dissociation of certain combustion products, as is implied by the user manual [113]:

> *The combustion process is calculated using gas and fuel composition data and the equations for chemical equilibrium, meaning that dissociation effects are accounted for.*

Thirdly, GSP also include a calculation of other species which form harmful emissions, such as NO_x, unburned hydrocarbons and soot. This is one of the reasons as to why one would include dissociation effects i.e. to determine accurately the species resulting from combustion. The accurate determination of emissions is dependent on a multitude of factors, and is not included here due to its complexities. The reader may refer to Reference [68] for more on the subject. Nonetheless, the additional emission species calculated also effect the overall thermodynamic properties of the gas mixture exiting the combustion chamber. In addition to the above, GSP includes further advanced capabilities which, for example, allow it to include the effects of compressibility and presence of liquid vapour in the combustion process (although these have been disabled for the validation purposes).

Therefore, it is evident that GSP's combustion model is far more sophisticated than the one presented in this section. However, this level of sophistication is not necessary for the purposes of this work, and any improvements in accuracy (pertaining to this particular work) do not warrant the time and effort required for its development. The discrepancies between the this model and GSP's, are primarily a result of the exclusion and inclusion of dissociation, respectively. However, in the context of gas turbine modelling dissociation effects are typically deemed negligible [130].

A comparison of the results produced by the two models, Figure E.23 may now be examined in context of the above differences. Figure E.23a, illustrates large errors in the prediction of the composition of common combustion products. These errors, although large, do not impact too significantly on the accuracy of the adiabatic flame temperature calculated (error well within 5%), as is illustrated in Figure E.23b.

The impact of dissociation on the discrepancy between the two models, is clearly evident from Figure E.23. Firstly, the error increases with an increase in fuel-air ratio. This is because the adiabatic flame temperature is higher for greater fuel-air ratios [7], thereby providing more energy for dissociation reactions to occur.

[7] It is noted that this is not always the case. If the combustion runs at to rich a mixture the combustion becomes largely incomplete, thereby reducing the temperature of the combustion products. However, it is very unlikely for a gas turbine engine to run a rich mixture.

Secondly, the error in adiabatic temperature becomes more positive with an increase in fuel-air ratio. This is because the greater dissociation at higher temperatures results in the greater 'absorbtion' of heat, and therefore the dissociated adiabatic temperature becomes lower than the non-dissociated adiabatic flame temperature for increases in fuel-air ratios.

Thirdly, there is a clear hysteresis in the error produced. This is because the validation test was performed for both an acceleration and deceleration. Therefore, although at the same fuel-air ratio, the inlet temperature to the combustion chamber was different for acceleration and deceleration. The difference in inlet temperature results in different levels of dissociation. In the combustion model developed in this section, the combustion chamber's inlet temperature has no effect on Equations E.18, E.22 or E.21 and so does not influence the composition of the combustion products. It is because of this that the hysteresis is so clearly evident in the error plots of Figure E.23.

In addition, Figure E.23 gives evidence of a discrepancy in the composition of air used. This is clearly illustrated by the positive error in presence of N_2 at very lean mixtures. This is because the model developed here makes use of apparent Nitrogen, and therefore it over predicts the amount of Nitrogen in the mixture by accounting for the presence of other species such as Argon.

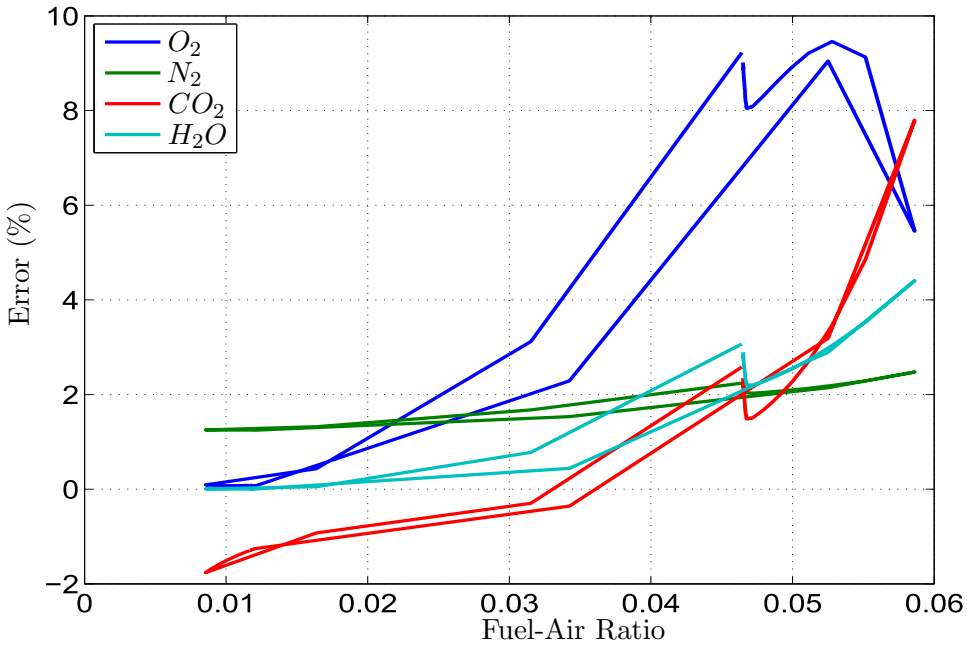
Nevertheless, the combustion model developed here is deemed to be satisfactory for the purposes of this work. This is because the adiabatic flame temperature calculated falls within a reasonable level of accuracy (within 5%), and it is this parameter which has the predominant effect of engine dynamics. The discrepancies in composition only really effect the prediction of emissions, which is not currently of interest to this research project.

Finally, it must be noted that results of pressure losses were not compared. This is because GSP allows the user to define a constant pressure loss value, which is exactly the method used in this model. Therefore, a validation test on pressure losses is unnecessary.

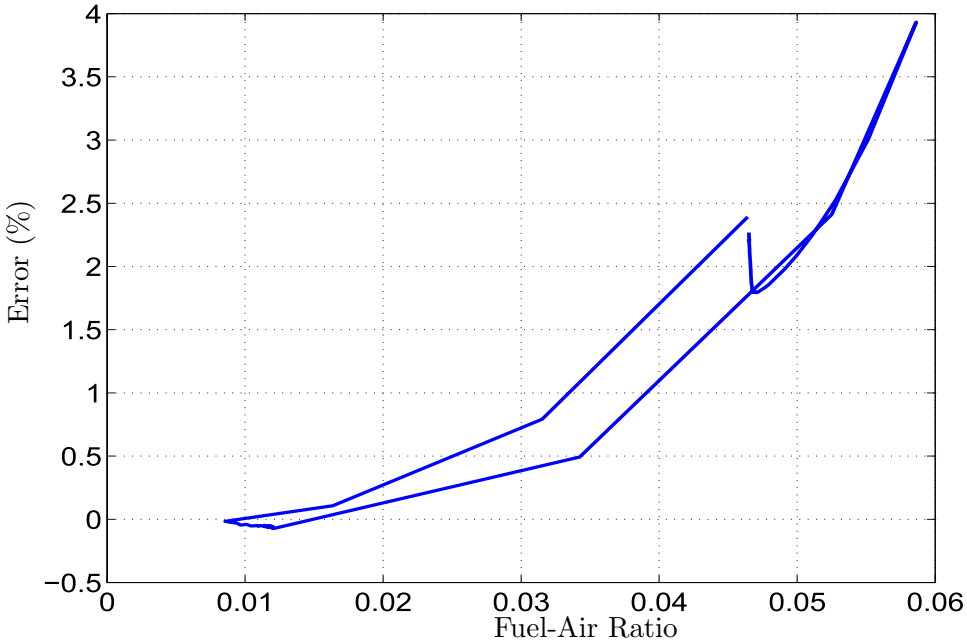
E.6 Turbines

E.6.1 Theory

The approach taken in calculating the performance and exit conditions of a turbine, is very similar to that of the compressor, and therefore the reader will be referred to Appendix E.3 to avoid repetition. The parameters (mass flow rate and isentropic efficiency) defining the turbines performance, are obtained from turbine maps (see Appendix D.2) given specified values of pressure ratio and corrected operating speed. The definition of isentropic efficiency is then used to calculate the temperature ratio across the turbine. The definition of the isentropic efficiency differs slightly from that of the



(a) Error of composition calculation of common combustion products



(b) Error of in calculated adiabatic flame temperature

Figure E.23: Error in combustion calculation error compared to GSP

compressor, and is a comparison of the actual power versus the ideal power delivered by the turbine, as in Equation E.27 obtained from Flack [33].

$$\eta_t = \frac{h_{ti} - h_{te}}{h_{ti} - h_{s,te}} \tag{E.27}$$

The temperature ratio, as with the compressor, is calculated using an iterative procedure that takes into account variable specific heats. This iterative procedure is essentially the same as that depicted in Figure E.7 for the compressor, except for Equation E.9 which is replace by Equation E.28 from Flack [33].

$$TR_t = \left(PR^{\frac{\gamma-1}{\gamma}} - 1\right)\eta_t + 1 \tag{E.28}$$

In addition, the power and torque delivered by the turbine is still given by Equations E.10 and E.11, respectively. However, it is important that attention be paid to the sign denoting the direction of that power, which is opposite to the compressor (i.e. the compressor absorbs power whereas the turbine delivers power).

In summary, as with the compressor, the turbine's exit thermodynamic properties, calculated from the theory above, is used to define the inlet boundary condition of the subsequent downstream component/s. The calculated power is used in determining torque imbalances between compressors and turbines fixed to a common spool, thus dictating rotor dynamics. Lastly the mass flow rate, determined from turbine maps, is required for determining the rates of change in pressure inside the volumes adjacent to the turbine, which is used in modelling the volume dynamics of the engine.

E.6.2 Simulink Model

The Simulink block developed for modelling turbine components is structurally very similar to that of the compressor, and is depicted in Figure E.24. Figure E.24a illustrates the highest level subsystem, which is used for interfacing with other engine component models. This block has input ports that take in the turbine's exit pressure, a bus signal from another upstream component and the speed of the spool to which the turbine is fixed. A new bus signal, containing all of the turbine's associated parameters, is then passed out of the system, along with the torque produced and mass flow rate.

Figure E.24b depicts the user-interface level of the Turbine subsystem. It is here that the user must define the turbine's design point, which is used to scale the generic turbine map [8]. The user is also required to allocate a station number (or name) to the turbine's data output. This name will be

[8] The user is also required to select a general map type suited to high, intermediate or low pressures. This is done so within the Embedded MATLAB Function in Figure E.24c

used export all the turbine's data to the MATLAB workspace. The design point is passed to the subsystem 'Turbine Calcs', along with the three signals passed to the high-level block (Figure E.24a).

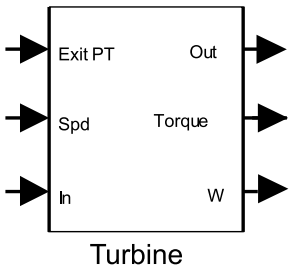
Finally, Figure E.24c illustrates the 'Turbine Calcs' subsystem of Figure E.24b. Here, the signals from the higher levels are past to an 'MATLAB Turbine Function'. The user is required to check that the bus signal selector (black bar on the left hand side) has assigned inlet gas composition, total pressure and temperature to the correct inlet ports of the MATLAB embedded function. The 'MATLAB Turbine Function' passes information from the Simulink environment to the appropriate MATLAB functions. This allows for complex iterative calculations to be conducted within a single time step of the Simulink environment (The embedded MATLAB function and associated routines are discussed in the next section). Unlike the compressor model, the turbine model uses a far simpler interpolation scheme when obtaining data from performance maps, and therefore there is no need to provide an estimate of the map's β value.

E.6.3 MATLAB Code

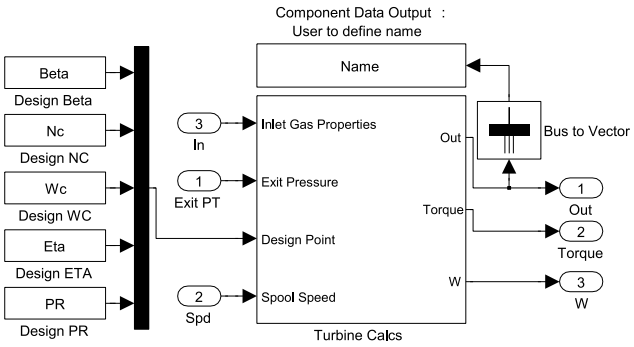
Again, the architecture of the turbine functions is very similar to that of the compressors. The actual turbine calculations are contained within MATLAB functions, and not as Simulink blocks. This is because, as previously mentioned, the turbine calculations require iterative procedures that must be completed within a single time step. This subsection describes the MATLAB functions developed to perform these calculations.

The first is an Embedded MATLAB Function, 'MATLAB Turbine Function', who's purpose is solely to facilitate communication of inputs and outputs between the Simulink and MATLAB environments. The contents of this function are presented in Code Listing E.8. It is here that the user is required to specify the map type to be used. An appropriate map may be selected for a low, intermediate or high pressure turbine. The main purpose of this embedded function is to call a subroutine, 'Turbine.m' presented in Code Listing E.9, that performs the relevant calculations in the MATLAB environment. This subroutine first calculates the pressure ratio across the turbine as well as the corrected speed according to the inlet total temperature. These two parameters are then passed to another subroutine, 'TurbMapsInterpPR.m', which is used to determine the corrected mass flow rate and isentropic efficiency at which the turbine operates [9]. 'Turbine.m', of Code Listing E.9, then calculates the mass flow rate through the turbine from the corrected mass flow rate interpolated from the turbine map in the previous subroutine call. Next, the pressure ratio, inlet total temperature and efficiency are passed to a subroutine 'TurbTherm.m' (Code

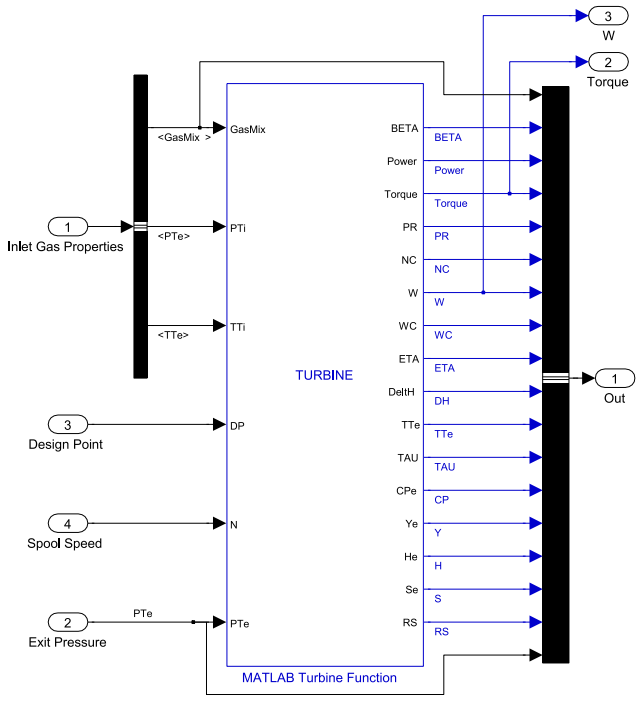
[9] This subroutine is detailed in Section D.2 and is found in Code Listing D.4



(a) High-level turbine block



(b) User-interface level



(c) Turbine calculations subsystem

Figure E.24: Turbine model developed in Simulink

Listing E.10), which implements the iterative procedure (detailed in Figure E.7, but adjusted for turbine in accordance with Section E.6.1) to calculate the total temperature (and subsequent thermodynamic properties) exiting the turbine. Finally, 'Turbine.m', calculates the power and torque delivered by the turbine. All the above mentioned parameters are then returned to the Simulink block through the embedded MATLAB function.

Code Listing E.8: Embedded MATLAB Function TURBINE

```
%=============================================================
%              TURBINE EMBEDDED MATLAB FUNCTION
%=============================================================
%
% NAME:          TURBINE
%
% AUTHOR:        D. Sanders, University of Cape Town
%
% DATE:          21 November 2012
%
% FUNCTION:      [BETA, Power, Torque, PR, NC, W, WC, ETA, ...
    DeltH, TTe, TAU, CPe, Ye, He, Se, RS] = TURBINE ...
    (GasMix, PTi, TTi, DP, N, PTe)
%
% PURPOSE:       Call turbine MATLAB routines
%
% INPUTS:
%
%   PTe          = Exit Pressure                  [Pa]
%   PTi          = Inlet Pressure                 [Pa]
%   DP           = Design Point Vector   [BETA NC WC ETA PR]
%   TTi          = Inlet Temperature              [K]
%   N            = Spool Speed                    [rpm]
%   GasMix       = Gas composition
%
% OUTPUTS:
%
%   BETA         = Map BETA value                 [-]
%   Power        = Required Power                 [W]
%   Torque       = Required Compressor Torque     [N.m]
%   PR           = Pressure Ratio                 [PR]
%   NC           = Corrected rotational speed     [rpm]
%   W            = Mass Flow Rate                 [kg/s]
%   WC           = Corrected Mass Flow Rate       [kg/s]
%   ETA          = Efficiency                     [-]
%   DeltH        = Enthalpy rise                  [J/kg]
%   TTe          = Exit Temperature               [K]
%   CPe          = Exit specific heat             [J/kg-K]
%   Ye           = Exit specific heat ratio       [-]
%   He           = Exit enthalpy                  [J/kg]
%   Se           = Exit entropy                   [J/kg-K]
%   Rs           = Specific Gas Constant          [J/kg-K]
%
```

```
42 % SUBROUTINES:  Turbine.m
43 %
44 % REFERENCES:
45 %
46 % For coding conventions, please refer to:
47 %
48 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
49 %   ver 1.5. October 2002.
50 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
51 %===============================================================
52
53 function [BETA, Power, Torque, PR, NC, W, WC, ETA, DeltH, ...
       TTe, TAU, CPe, Ye, He, Se, RS] = TURBINE (GasMix, PTi, ...
       TTi, DP, N, PTe)
54
55 %---------------------------------------------------------------
56 %                        TURBINE MAP SELECTION
57 %---------------------------------------------------------------
58
59 % Select general map to represent turbine through ...
       appropriate scaling: mapType = 'LPT' or 'LPT' or 'HPT'
60 mapType = 'HPT';
61
62 %---------------------------------------------------------------
63 %               VARIABLE INITIALISATION AND FUNCTION CALL
64 %---------------------------------------------------------------
65
66 % MATLAB function to be called
67 eml.extrinsic('Turbine');
68
69 % Initialise output variables
70 Power = 0; Torque=0; PR=0; NC=0; BETA=0; W=0; WC=0; ETA=0; ...
       DeltH=0; TTe=0; TAU=0; CPe=0; Ye=0; He=0; Se=0; RS=0;
71
72 % Call MATLAB function to calculate outputs
73 [Power, Torque, PR, NC, BETA, W, WC, ETA, DeltH, TTe, TAU, ...
       CPe, Ye, He, Se, RS] = Turbine (mapType, DP, PTe, N, ...
       PTi, TTi, GasMix);
```

Code Listing E.9: MATLAB Function Turbine.m: For calculating turbine thermodynamics

```
1  %===============================================================
2  %                     TURBINE CALCULATIONS
3  %===============================================================
4  %
5  % NAME:          Turbine.m
6  %
7  % AUTHOR:        D. Sanders, University of Cape Town
8  %
9  % DATE:          17 July 2012
10 %
```

```
11 % FUNCTION:     [Power, Torque, PR, NC, BETA, W, WC, ETA, ...
       DeltH, TTe, TAU, CPe, Ye, He, Se, RS] = Turbine ...
       (mapType, DP, PTe, N, PTi, TTi, GasMix)
12 %
13 % PURPOSE:      Calculate turbine operating conditions
14 %
15 % INPUTS:
16 %
17 %   mapType     = type of map to be used (string):
18 %                 low intermediate or high
19 %                 'LPT', 'IPT' or 'HPT'
20 %   PTe         = Exit Pressure                 [Pa]
21 %   PTi         = Inlet Pressure                [Pa]
22 %   DP          = Design Point Vector   [BETA NC WC ETA PR]
23 %   TTi         = Inlet Temperature             [K]
24 %   N           = Spool Speed                   [rpm]
25 %   GasMix      = Gas Composition       [FUEL O2 N2 CO2 H2O]
26 %
27 % OUTPUTS:
28 %
29 %   BETA        = Map BETA value                [-]
30 %   Power       = Delivered Power               [W]
31 %   Torque      = Delivered Torque              [N.m]
32 %   PR          = Pressure Ratio                [PR]
33 %   NC          = Corrected rotational speed    [rpm]
34 %   W           = Mass Flow Rate                [kg/s]
35 %   WC          = Corrected Mass Flow Rate      [kg/s]
36 %   ETA         = Efficiency                    [-]
37 %   DeltH       = Enthalpy rise                 [J/kg]
38 %   TTe         = Exit Temperature              [K]
39 %   CPe         = Exit specific heat            [J/kg-K]
40 %   Ye          = Exit specific heat ratio      [-]
41 %   He          = Exit enthalpy                 [J/kg]
42 %   Se          = Exit entropy                  [J/kg-K]
43 %   Rs          = Specific Gas Constant         [J/kg-K]
44 %
45 % SUBROUTINES:  TurbMapsInterpPR.m and TurbTherm.m
46 %
47 % REFERENCES:
48 %
49 % For coding conventions, please refer to:
50 %
51 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
52 %   ver 1.5. October 2002.
53 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
54 %===============================================================
55
56 function [Power, Torque, PR, NC, BETA, W, WC, ETA, DeltH, ...
       TTe, TAU, CPe, Ye, He, Se, Rs] = Turbine (mapType, DP, ...
       PTe, N, PTi, TTi, GasMix)
57
58 %% Turbine Map Handling
59
60 % Calculate Pressure Ratio
```

```
61 PR = PTe / PTi;
62
63 % Transpose DP vector for use in interpolation routine
64 DP = DP';
65
66 % Calculate NC
67 NC = N / sqrt(TTi / 288.15);
68
69 % Map Interpolation
70 [BETA WC ETA] = TurbMapsInterpPR (NC, PR, mapType, DP);
71
72 % Calculate mass flow rate
73 W = WC * (PTi/101.325e3) / sqrt(TTi / 288.15);
74
75 %% Thermodynamic Calculations:
76 [DeltH, TTe, TAU, CPe, Ye, He, Se, Rs] = TurbTherm (PR, ...
       TTi, ETA, GasMix);
77
78
79 %% Power & Torque Calculations
80
81 % Power
82 Power = DeltH * W;
83
84 % Torque
85 Torque = (Power * 60 / (2*pi*N));
86
87 return
```

Code Listing E.10: MATLAB Function TurbTherm.m: For implementing Figure E.7

```
1  %=============================================================
2  %            TURBINE THERMODYNAMIC CALCULATIONS
3  %=============================================================
4  %
5  % NAME:         TurbTherm.m
6  %
7  % AUTHOR:       D. Sanders, University of Cape Town
8  %
9  % DATE:         17 July 2012
10 %
11 % FUNCTION:     [DeltH, TTe, TAU, CPe, Ye, He, Se, Rs] = ...
       TurbTherm (PR, TTi, ETA, GasMix)
12 %
13 % PURPOSE:      Calculate turbine exit total temperature ...
       and corresponding thermodynamic properties
14 %
15 % INPUTS:
16 %
17 %   TTi         = Inlet Temperature             [K]
18 %   PR          = Pressure Ratio                [-]
```

```
19 %   ETA         = Efficiency                    [%/100]
20 %   GasMix      = Gas composition
21 %
22 % OUTPUTS:
23 %
24 %   DeltH       = Enthalpy rise                 [J/kg]
25 %   TTe         = Exit Temperature              [K]
26 %   TUA         = Temperature ratio             [-]
27 %   CPe         = Exit specific heats           [kJ/kg-K]
28 %   Ye          = Exit specific heats ratio     [-]
29 %   He          = Exit Enthalpy                 [kJ/kg]
30 %   Se          = Exit Entropy                  [kJ/kg-K]
31 %   Rs          = Gas constant                  [kJ/kg-K]
32 %
33 % SUBROUTINES:  CombTherm.m
34 %
35 % REFERENCES:
36 %
37 %   Flack, R. "Fundamentals of jet propulsion with ...
       applications" Cambridge Univ Pr, 2005, 17
38 %
39 % For coding conventions, please refer to:
40 %
41 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
42 %   ver 1.5. October 2002.
43 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
44 %===============================================================
45
46 function [DeltH, TTe, TAU, CPe, Ye, He, Se, Rs] = TurbTherm ...
       (PR, TTi, ETA, GasMix)
47
48 %% Thermodynamic Calculations:
49
50 % Specify specific heat ratio accuracy tolerence
51 ytol = 0.0001;
52
53 % Initialise specific heats estimate
54 [CP, Y, H, S, Rs] = CombTherm (TTi, TTi, GasMix);
55 deltY = 1;
56
57 % Inlet enthalpy for Work calcs
58 Hi = H;
59
60 % Calculate Tte using cp, y at average component temp
61 while deltY > ytol
62
63     % Calculate Temperature Ratio
64     TAU = ( ((PR)^((Y-1)/Y) - 1)*ETA ) + 1;
65     TTe = TAU * TTi;
66
67     % Calculate average temperature and subsequent specific ...
           heat ratio
68     TTm = (TTe + TTi)/2;
69     [CPm, Ym, Hm, Sm, Rs] = CombTherm (TTm, TTm, GasMix);
```

```
70
71      deltY = abs(Ym - Y);
72      Y = Ym;
73  end
74
75  % Use final specific heat value to calculate exit temperature
76  TAU = ( ((PR)^((Y-1)/Y) - 1)*ETA ) + 1;
77  TTe = TAU * TTi;
78  [CPe, Ye, He, Se, Rs] = CombTherm (TTe, TTe, GasMix);
79
80  % Calculate Energy extracted
81  DeltH = -(He - Hi);
82
83  return
```

E.6.4 Validation

The composition of the gases passing through the turbines are based on the products of combustion. This makes validating the turbine model difficult, because (as has been discussed in Section E.5 on combustion) the gas model developed for this work does not support all the species that GSP does. As a result, the turbine model cannot accommodate the same gas composition that GSP's turbine have to accommodate, and therefore the two model's are subject to different boundary conditions. In addition, the turbine model cannot be run in isolation as it does require an input of the gas composition used. Therefore in order to allow for some validation, the turbine model was run in conjunction with the combustion model. The combustion model is used only to approximate the composition of the gases passing through the turbines. In so doing, the turbine model inherits the errors of the combustion model, thereby compromising the validation results. Nonetheless, validation tests were run on the turbine model and are documented here.

The turbine model was tested for the same transient conditions as those presented in Section D.2.5. The validation tests were performed for both low and high pressure turbines, who's transient trajectories are depicted in Figures D.34a and D.35a respectively. For this validation, only the parameters of torque, mass flow rate and exit total temperature were considered. This is because torque effects the rotor dynamics, mass flow rate effects the volume dynamics and exit temperature influences the operation of downstream components. The results of the validation are depicted in Figures E.25 and E.26 for the low and high pressure turbines respectively.

It is very difficult to interpret these results and the source of the errors. This is because the cause of the errors are related to the combustion model (in addition to any from the turbine model), and transients introduce different fuel-air ratios. The different fuel-air ratios result in different compositions of combustion products. An error in this composition (compared to GSP) varies with the fuel-air ratios and therefore transients, as has been

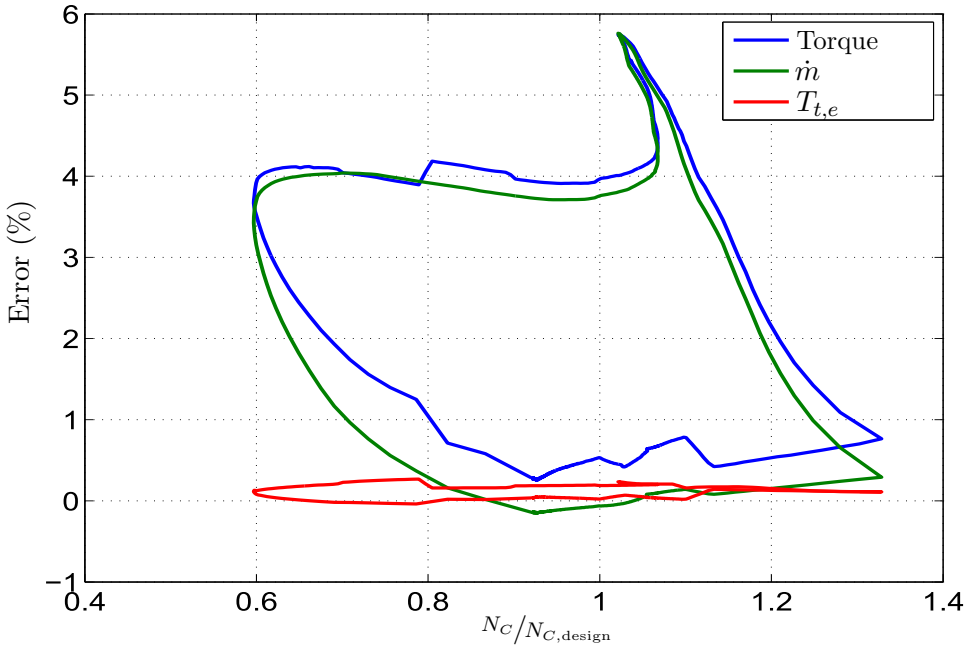


Figure E.25: Low pressure turbine: Validation of thermodynamic model compared to GSP

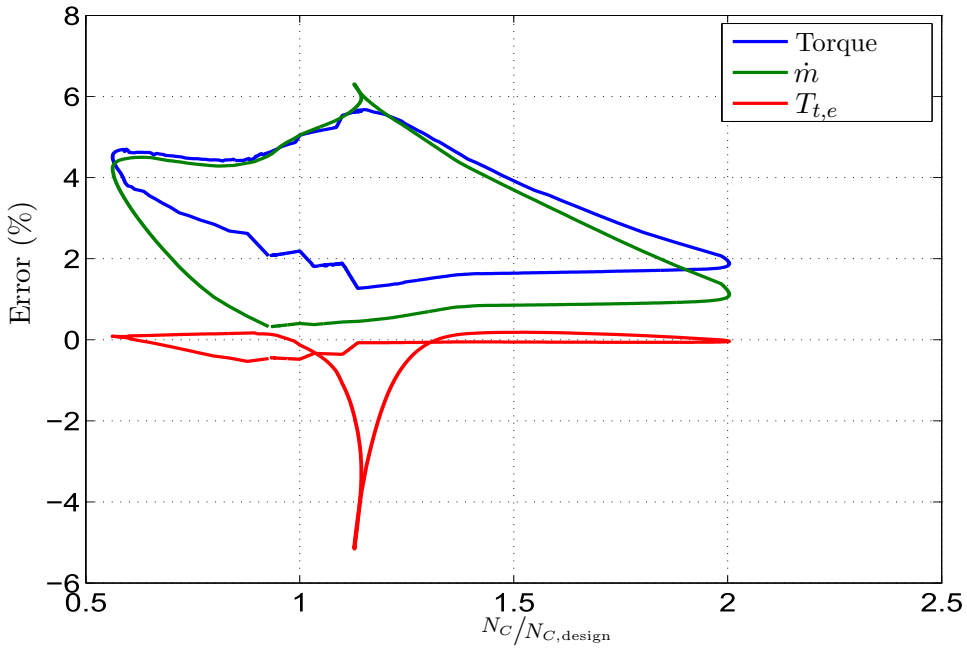


Figure E.26: High pressure turbine: Validation of thermodynamic model compared to GSP

demonstrated in Section E.5. Therefore, one would need to filter off the combustion errors (related to transient conditions) to reveal the error inherent in the turbine model. However, this is a rather complex and lengthy process and, due to the accuracy achieved, is not deemed necessary. The errors of Figures E.25 and E.26 seldom exceed 5%, and therefore it is likely that these errors would reduce to well within 5% , if one were to filter off the combustion related errors. Therefore the turbine model developed her is deemed valid relative to GSP software.

E.7 Nozzles

The purpose of a nozzle is to increase the kinetic energy (velocity) of the exhaust gases, thereby imparting momentum to the engine structure and airframe (i.e. thrust). Typically there are two types of nozzles which may be found on a jet engine (convergent and convergent-divergent), and either of these may be used to exhaust the primary or secondary (bypass) flows. In addition, these nozzle types may have variable or fixed geometries. However, for the turbofan engine modelled in this work, both primary and secondary nozzles are considered to be fixed-area convergent nozzles, and therefore convergent-divergent and variable geometry nozzles are not discussed in the sections that follow. The theory presented here on fixed-area convergent nozzles, follows closely to that described by Flack [33].

E.7.1 Theory

A nozzle works in an opposite way to a diffuser, in that its purpose is to convert static enthalpy to dynamic enthalpy. This expansion process is assumed to be adiabatic and with no work transfer. A measure of how well the nozzle achieves the above purpose, is represented by its isentropic efficiency:

$$\eta_n = \frac{h_{ti} - h_e}{h_{ti} - h_{s,e}} \tag{E.29}$$

The velocity at which the gases are expelled from the nozzle, is required to calculate the amount of thrust generated. The amount of enthalpy converted from static to dynamic, defines the increase in velocity of gases flowing through the nozzle, an thus the velocity of the gases exiting the nozzle is obtained from Equation E.30.

$$u_e = \sqrt{2\left(h_{ti} - h_e\right)} = \sqrt{2\eta_n\left(h_{ti} - h_{s,e}\right)} \tag{E.30}$$

Because the expansion through the nozzle is assumed to be adiabatic with no work transferred, it is assumed that the specific heats are the same at inlet and exit, and therefore constant everywhere within the nozzle. Thus,

the nozzle expansion process takes on the specific heat value (and therefore specific heats ratio) from its preceding component. This is a reasonable assumption because the nozzle operates adiabatically and with no work. Hence Equation E.30 becomes:

$$u_e = \sqrt{2\eta_n c_p \left(T_{ti} - T_{s,e}\right)} \tag{E.31}$$

If the process is assumed to be isentropic, then the stagnation isentropic pressure and temperature are equal to the stagnation inlet pressure and temperature:

$$P_{s,te} = P_{ti} \quad \& \quad T_{s,te} = T_{ti} \tag{E.32}$$

The isentropic exit static temperature may be obtained from Equation E.33, where the exit Mach number (at isentropic conditions) is obtained from Equation E.34.

$$T_{s,e} = \frac{T_{ti}}{1 + \frac{\gamma-1}{2} M_{s,e}^2} \tag{E.33}$$

$$M_{s,e} = \sqrt{\frac{2}{\gamma - 1}\left[\left(\frac{P_{s,te}}{P_a}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]} \tag{E.34}$$

The exit velocity and static temperature (assuming that the isentropic efficiency is known) is then calculated by solving Equations E.31 to E.34 simultaneously.

However, it is not only the velocity of the exhaust gases which influence the thrust achieved. The exit static pressure also influences the amount of momentum is imparted by the gases to the nozzle, and is obtained from Equation E.35.

$$\frac{P_e}{P_{ti}} = \left(\frac{\frac{T_e}{T_{ti}} - 1}{\eta_n} + 1\right)^{\frac{\gamma}{\gamma-1}} \tag{E.35}$$

Subsequently, the exit static pressure may be used to calculate the density of the exhaust gases from the ideal gas law, Equation E.36, which allows for the calculation of the mass flow rate at which the gasses are expelled from the nozzle, Equation E.37.

$$\rho_e = \frac{P_e}{RT_e} \tag{E.36}$$

$$\dot{m}_e = \left(\rho A u\right)_e \tag{E.37}$$

However, sometimes the nozzle will operate at sonic conditions. When sonic conditions are reached, the nozzle is said to be choked. This is because

the nozzle begins to behave more like diffuser, when sonic conditions are exceeded, thereby capping any further expansion of the gases (i.e. places a limit on the conversion of static enthalpy to dynamic enthalpy). When this occurs, the Mach number (at the point of choking) may be assumed to equal unity (in actuality, this would be slightly below unity but is a close enough approximation [33]) and the corresponding static sonic pressure and temperature are defined by Equations E.38 and E.39, respectively (Obtained from Equations E.35 and E.33, where the Mach number is of unity).

$$\frac{P_{\text{sonic}}}{P_{ti}} = \left[1 + \frac{1-\gamma}{\eta_n(1+\gamma)}\right]^{\frac{\gamma}{\gamma-1}} \tag{E.38}$$

$$\frac{T_{\text{sonic}}}{T_{ti}} = \frac{2}{1+\gamma} \tag{E.39}$$

These sonic conditions occur if the atmospheric static pressure is less than that of the sonic condition (i.e. $P_{atm} < P_{\text{sonic}}$) [33]. In this case the exit nozzle static temperature and pressure are assumed to equal their sonic equivalents, and the exit velocity, density, mass flow rate and thrust are obtained from Equations E.31 and E.36 to E.44, based on these values.

Then, for choked or un-choked conditions, one may want to know the stagnation pressure (Equation E.40) and temperature (Equation E.41) at the nozzle's exit (Although these parameters to not influence any other components or the thrust achieved). Thereby defining the pressure (Equation E.42) and temperature (E.43) ratios at which the nozzle operates.

$$\frac{P_{te}}{P_e} = \left[1 + \frac{\gamma-1}{2{M_e}^2}\right]^{\frac{\gamma}{\gamma-1}} \tag{E.40}$$

$$\frac{T_{te}}{T_e} = \left[1 + \frac{\gamma-1}{2{M_e}^2}\right] \tag{E.41}$$

$$PR = \frac{P_{te}}{P_{ti}} \tag{E.42}$$

$$TR = \frac{T_{te}}{T_{ti}} \tag{E.43}$$

The calculations above, are dependent on a known efficiency and exhaust area. The isentropic efficiency at which a nozzle operates, will clearly be dependent on the boundary conditions of operation. The manner by which efficiency varies with changes in boundary conditions, may be obtained from test rig data or alternatively higher fidelity models (e.g. CFD simulations). However, this work makes the simple assumption that the nozzle efficiency remains constant across the engine's envelope of operation. Unlike the diffuser, which operates in an adverse pressure gradient (e.g. causing flow

separation), there is little reason for the efficiency to vary drastically, and therefore this assumption is deemed reasonable.

Due to engine data typically being kept proprietary by engine manufacturers, it is difficult to obtain information on the actual exhaust nozzle area. Instead, the engine's design conditions are used to determine an effective exhaust throat area. This is done using the above theory, but assuming that the exhaust gas is expanded ideally to atmospheric static pressure, when the engine is operating at its design conditions. Thereafter, this design effective throat area is held constant for any off-design transient simulations of the engines performance.

Furthermore, losses due to pressure stacking may occur between the exit of the component preceding the nozzle and the nozzle inlet [113]. This loss may be included in the Inter-Component Volume (see Section E.7.4), which is wedged between the nozzle and its preceding component, and used for determining volume dynamics. The pressure stacking loss is left to the user to define, and is held constant across the operating envelope of the engine.

Finally, the results of the above calculations may be used to determine the thrust produced at the nozzle, which is obtained from Equation E.44[10].

$$\mathrm{F} = u_e \times \dot{m}_e \tag{E.44}$$

E.7.2 Simulink Model

The theory presented above has been developed into a Simulink block, which is depicted in Figure E.27. The actual calculations are performed by a MATLAB routine, which is discussed in the following subsection.

Figure E.27a illustrates the highest level of the nozzle Simulink block developed. This level is used for interfacing with other engine components, and takes in a bus signal of the exit conditions of its preceding component (this will typically be an Inter-Component Volume block) as well as a signal specifying atmospheric static pressure.

Figure E.27b illustrates the user-interface level of the nozzle Simulink block developed. Here, the user is required to specify the exhaust nozzle area (if known, otherwise this may be obtained from an engine design point calculation) and the isentropic efficiency at which the nozzle operates. Both these values are assumed to remain constant after specification. The user is also required to name the data output from this block. The subsequent nozzle exit data is then recorded to the MATLAB workspace under this name.

Figure E.27c, depicts the 'Nozzle Calcs' subsystem of Figure E.27b. Here, the relevant inputs are extracted from the bus signals and, along with the specified values of exit area and isentropic efficiency, are passed to

[10] It is noted that this is purely the thrust that is generated at the nozzle, and does not account for ram drag. The ram drag is included in the intake model

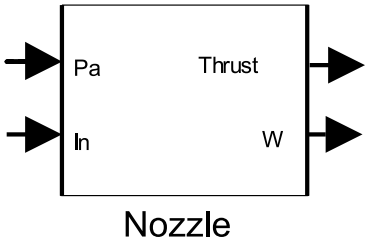
an 'Embedded MATLAB Nozzle Function'. This embedded function is used to pass inputs and outputs between the SImulink and MATLAB environments, via function calls (the actual nozzle calculations are performed by a MATLAB function).

E.7.3 MATLAB Code

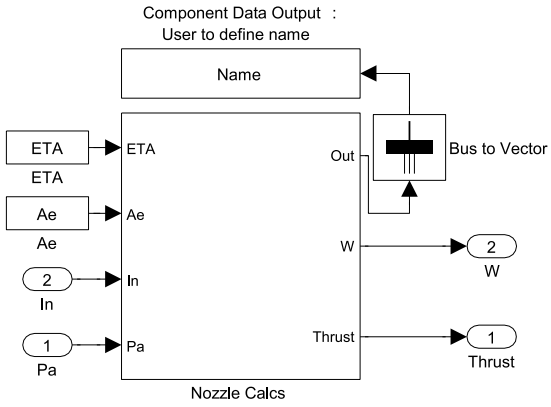
The nozzle calculations are primarily performed by a MATLAB function, and not by Simulink blocks. An embedded MATLAB function is used to call the MATLAB function from the Simulink environment, passing between them the inputs and outputs of the calculations. The embedded MATLAB function of Figure E.27c is presented in Code Listing E.11, and is simply used to call a MATLAB function 'Nozzle.m', which is presented in Code Listing E.12. 'Nozzle.m' is contains all the theory presented above, and is simple enough not to warrant any further explanation (The reader is kindly asked to refer to the code's comments for explanations where necessary).

Code Listing E.11: Embedded MATLAB Function NOZZLE

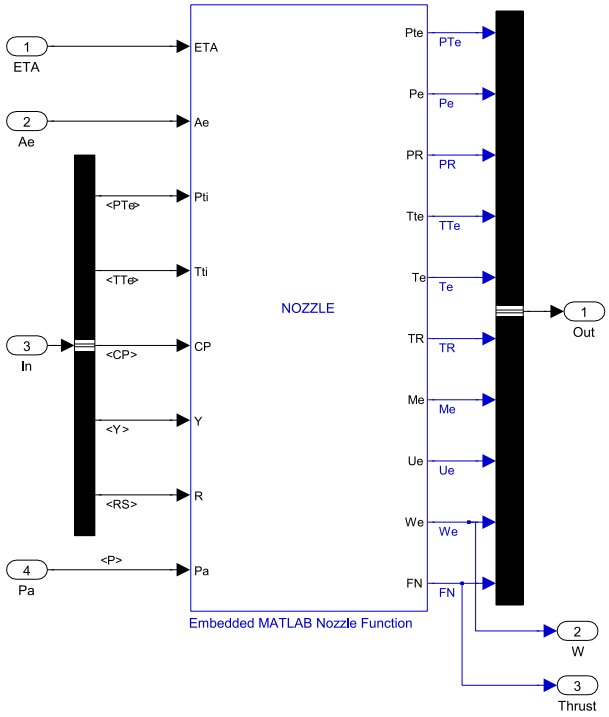
```
%=============================================================
%                        NOZZLE BLOCK
%=============================================================
%
% NAME:         NOZZLE
%
% AUTHOR:       D. Sanders, University of Cape Town
%
% DATE:         21 November 2012
%
% FUNCTION:     [Pte, Pe, PR, Tte, Te, TR, Me, Ue, We, FN] ...
    = NOZZLE (ETA, Ae, Pti, Tti, CP, Y, R, Pa)
%
% PURPOSE:      Call nozzle.m function to perform nozzle ...
    calculations
%
% INPUTS:
%
%   PTi         = Inlet Pressure                [Pa]
%   TTi         = Inlet Temperature             [K]
%   CP          = Specific Heat                 [J/kg-K]
%   Y           = Specific Heat Ratio           [-]
%   Rs          = Specific Gas Constant         [J/kg-K]
%   Pa          = Free Stream Pressure          [Pa]
%   ETA         = Nozzle Efficiency             [%/100]
%   Ae          = Design Effective exit area    [m^2]
%
% OUTPUTS:
%
%   Pte         = Exit stagnation pressure      [Pa]
%   Pe          = Exit static pressure          [Pa]
```



(a) High-level nozzle block



(b) User-interface level



(c) Nozzle calculations subsystem

Figure E.27: Nozzle model developed in Simulink

```
30 %   PR          = Pressure ratio                [-]
31 %   Tte         = Exit stagnation temperature   [K]
32 %   Te          = Exit static temperature       [K]
33 %   TR          = Temperature ratio             [-]
34 %   Me          = Exit Mach number              [-]
35 %   Ue          = exit velocity                 [m/s]
36 %   We          = Exit mass flow rate           [kg/s]
37 %   FN          = Thrust                        [N]
38 %
39 % SUBROUTINES:  Nozzle.m
40 %
41 % REFERENCES:
42 %
43 % For coding conventions, please refer to:
44 %
45 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
46 %   ver 1.5. October 2002.
47 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
48 %===========================================================
49
50 function [Pte, Pe, PR, Tte, Te, TR, Me, Ue, We, FN] = ...
       NOZZLE (ETA, Ae, Pti, Tti, CP, Y, R, Pa)
51
52 %-----------------------------------------------------------
53 %               VARIABLE INITIALISATION AND FUNCTION CALL
54 %-----------------------------------------------------------
55
56 % MATLAB function to be called
57 eml.extrinsic('Nozzle');
58
59 % Initialise output variables
60 Pte = 0; Pe = 0; PR = 0; Tte = 0; Te = 0; TR = 0; Me = 0; ...
       Ue = 0; We = 0; FN = 0;
61
62 % Call MATLAB function to calculate outputs
63 [Pte, Pe, PR, Tte, Te, TR, Me, Ue, We, FN] = Nozzle (ETA, ...
       Ae, Pti, Tti, CP, Y, R, Pa);
```

Code Listing E.12: MATLAB Function Nozzle.m: For performing nozzle calculations

```
1 %===========================================================
2 %                        NOZZLE CALCULATIONS
3 %===========================================================
4 %
5 % NAME:         Nozzle.m
6 %
7 % AUTHOR:       D. Sanders, University of Cape Town
8 %
9 % DATE:         21 November 2012
10 %
```

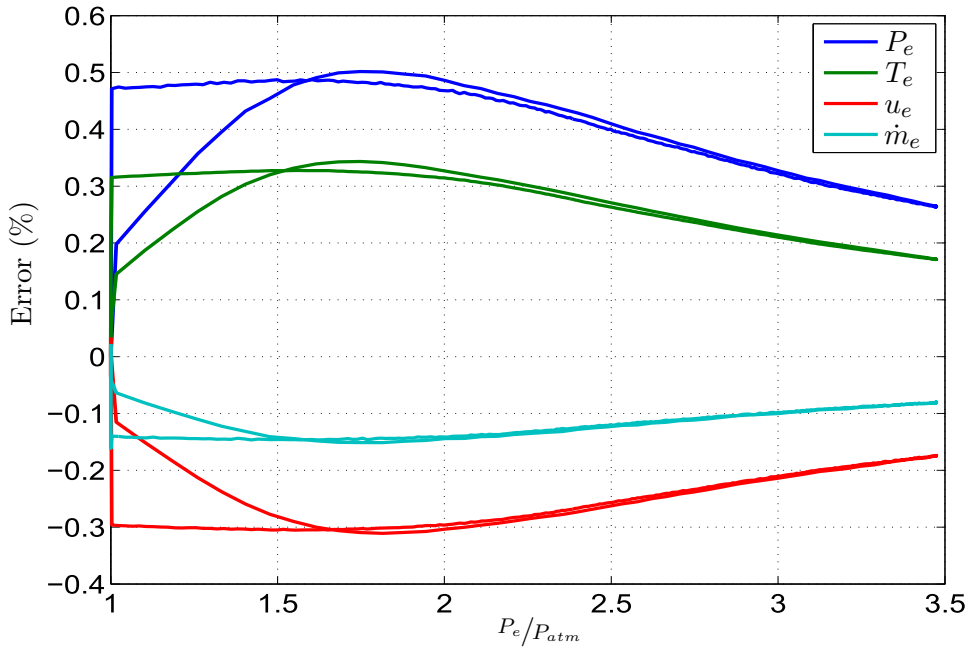
```
11 % FUNCTION:     [Pte, Pe, PR, Tte, Te, TR, Me, Ue, We, FN] ...
       = NOZZLE (ETA, Ae, Pti, Tti, CP, Y, R, Pa)
12 %
13 % PURPOSE:      Call nozzle.m function to perform nozzle ...
       calculations
14 %
15 % INPUTS:
16 %
17 %   PTi         = Inlet Pressure                [Pa]
18 %   TTi         = Inlet Temperature             [K]
19 %   CP          = Specific Heat                 [J/kg-K]
20 %   Y           = Specific Heat Ratio           [-]
21 %   Rs          = Specific Gas Constant         [J/kg-K]
22 %   Pa          = Free Stream Pressure          [Pa]
23 %   ETA         = Nozzle Efficiency             [%/100]
24 %   Ae          = Design Effective exit area    [m^2]
25 %
26 % OUTPUTS:
27 %
28 %   Pte         = Exit stagnation pressure      [Pa]
29 %   Pe          = Exit static pressure          [Pa]
30 %   PR          = Pressure ratio                [-]
31 %   Tte         = Exit stagnation temperature   [K]
32 %   Te          = Exit static temperature       [K]
33 %   TR          = Temperature ratio             [-]
34 %   Me          = Exit Mach number              [-]
35 %   Ue          = exit velocity                 [m/s]
36 %   We          = Exit mass flow rate           [kg/s]
37 %   FN          = Thrust                        [N]
38 %
39 % SUBROUTINES:  none
40 %
41 % REFERENCES:
42 %
43 %   Ronald D Flack. Fundamentals of Jet Propulsion with ...
       Applications. Number ISBN-13 978-0-521-81983-1. ...
       Cambridge University Press, New York, 2005.
44 %
45 % For coding conventions, please refer to:
46 %
47 %   Johnson, Richard. "MATLAB Programming Style Guidlines".
48 %   ver 1.5. October 2002.
49 %   http://www.mathworks.com/matlabcentral/fileexchange/2529
50 %===============================================================
51
52 function [Pte, Pe, PR, Tte, Te, TR, Me, Ue, We, FN] = ...
       Nozzle (ETA, Ae, Pti, Tti, CP, Y, R, Pa)
53
54 %% Ideal nozzle calculations
55
56 % Isentropic
57 Ptei = Pti;
58 % Ptei = Pa
59 Ttei = Tti;
```

```
60
61 % Assuming Pa = Pe using Flack eq's H.2.13; H.2.9;
62 Mei = sqrt( 2/(Y-1)*( (Ptei/Pa)^((Y-1)/Y) - 1) );
63 Tei = Ttei/(1 + (Y-1)/2*Mei^2);
64
65 %% Non-ideal calculations (Adiabatic)
66
67 % Flack Eq's 3.2.35; 3.2.32; 3.2.36; 3.2.38; H.2.13; H.2.9
68 Ue = sqrt( 2*CP*ETA*(Tti-Tei) );
69 Te = Tti - ETA*(Tti - Tei);
70 Me = sqrt( (Tti/Te - 1) / ((Y-1)/2) );
71 Pe = Pti * ( (Te/Tti - 1 + ETA) / ETA )^( Y/(Y-1) );
72
73 % Check for choked nozzle
74
75 % Flack 3.2.39
76 Pc = Pti * (1 + (1-Y)/ETA/(1+Y))^(Y/(Y-1));
77
78 if (Pa < Pc)
79 warning('nozzle choking')
80     % In actuality for a converging nozzle with n<1 the Me is
81     % slightly <1. Assmuption to use unity because effect on
82     % thrust and TSFC (<0.1%) is minimal
83     Me = 1;
84
85     % No longer assumes Pe = Pa
86     Pe = Pc;
87
88     % Flack 3.2.40; H.2.13; H.2.9;
89     Te = 2 * Tti / (Y+1);
90
91     % Flack 3.2.34;
92     Ue = sqrt( 2*CP*(Tti - Te)); % m/s
93
94 end
95
96 %% Stagnation exit temp and pressure
97     Pte = Pe * (1 + (Y-1)/2*Me^2)^(Y/(Y-1));
98     Tte = Te * (1 + (Y-1)/2*Me^2);
99     PR = Pte / Pti;
100    TR = Tte / Tti;
101
102 %% Calculate Nozzle exit conditions;
103
104 % From Ideal gas law
105 rho = Pe /(R*Te); % kg/m^3
106
107 % From Continuity
108 We = Ae * (rho*Ue);   %
109
110 %% Calculate Thrust
111 FN = (We*Ue) + Ae*(Pe - Pa);
112
113 return
```

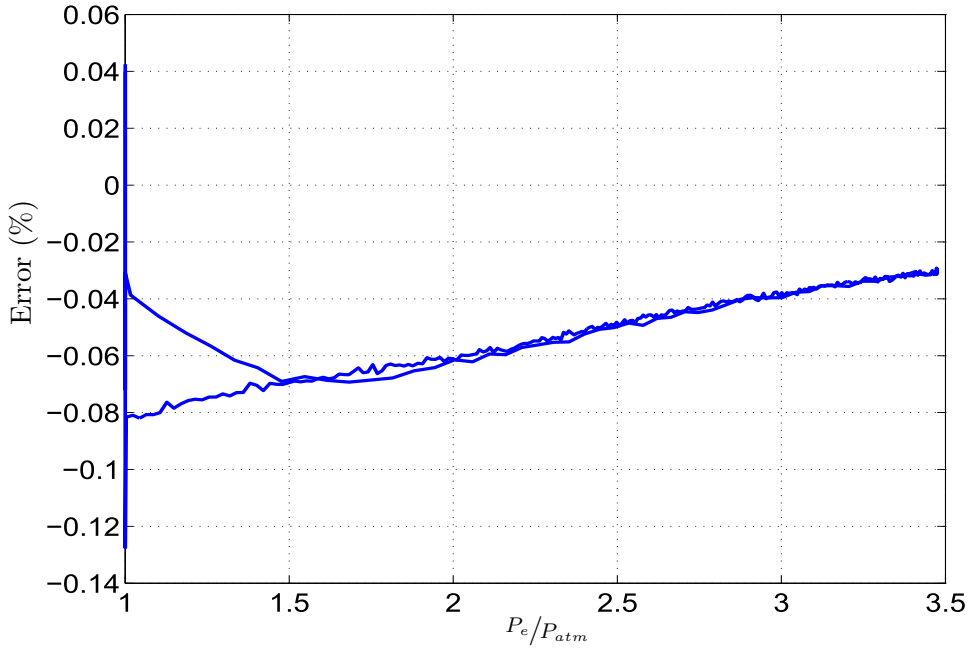
E.7.4 Validation

'Nozzle.m' was tested against results from GSP under transient operation at cruise conditions. The same transient condition that has been used to validate all preceding components has been utilised for this validation too. Validation tests were performed for both primary (core flow) and secondary (bypass flow) nozzles. The results of which are depicted in Figures E.28 and E.29, respectively.

Figures E.28a and E.29a, show how accurately the nozzle model calculates the exit parameters; static pressure and temperature, velocity and mass flow rate. The corresponding thrust produced at each nozzle is depicted by Figures E.28b and E.29b, respectively. These figures illustrate the error as a function of the exit static pressure normalised by the static atmospheric pressure. This format of presentation was utilised to gain insight as to what effect the level to which the gas is under-expanded, has on the error produced. It is difficult to identify a particular trend from the results of the primary nozzle, whereas the secondary nozzle suggests that the less expanded the gas the greater the error. Nonetheless, the errors are all significantly small enough, not to warrant any further speculation about there source. One comment is necessary just to explain the hysteresis evident in most of the figures. Hysteresis is present because the transient investigated include accelerations and decelerations, which occur along different trajectories. Therefore, although the normalised exit static pressure is equal, the inlet temperature (and exit static temperature) is not (comparing acceleration and deceleration paths). It is these differences in temperatures which result in different exit conditions, thus causing hysteresis. The hysteresis is more prominent in the primary nozzle, simply because it operates under a larger temperature range than the secondary nozzle (as it exhausts combustion products, whereas the secondary exhausts air). On the whole, the accuracy is satisfactory, and the model is thereby deemed to be valid.

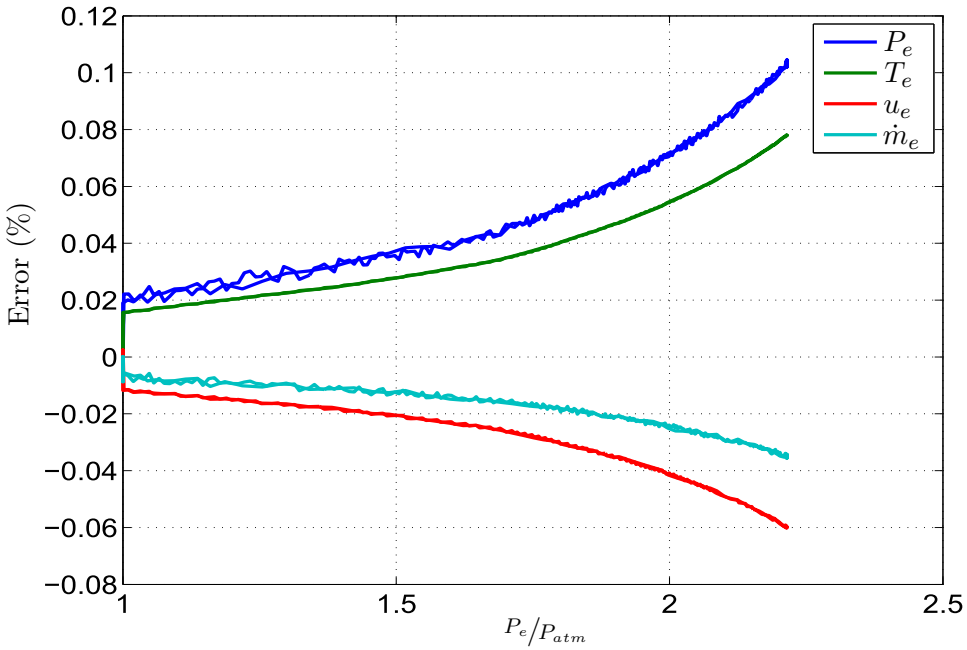


(a) Error in the calculation of nozzle exit parameters

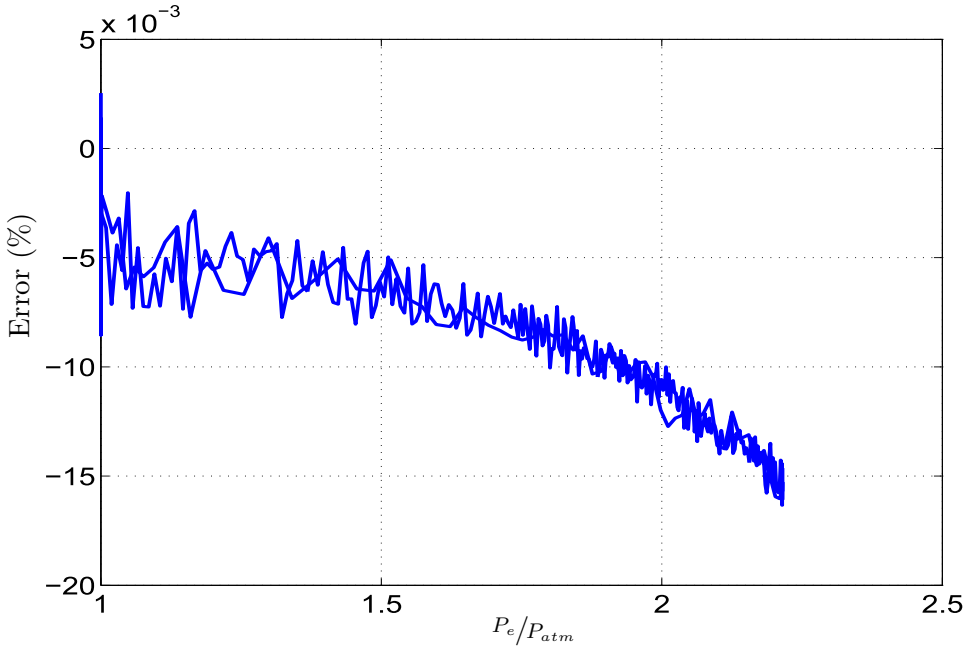


(b) Error in the calculation of thrust

Figure E.28: Error in primary nozzle calculations, compared to GSP



(a) Error in the calculation of nozzle exit parameters



(b) Error in the calculation of thrust

Figure E.29: Error in secondary nozzle calculations, compared to GSP

References

[1] Tony Tyler. Annual Review 2013. Technical report, The International Air Transport Association, June 2013.

[2] ICAO. Annual Report of the Council. Report, International Civil Aviantion Organisation, 2012.

[3] Anming Brueckner, Jan Keith; Zhang. Airline Emission Charges: Effects on Airfares, Service Quality, and Aircraft Design. *CESifo working paper*, (2547), February 2009.

[4] Fairuz I. Romli and Mohd Syahidie Kamaruddin. Preliminary Study of Emissions Regulation Effects on Future Commercial Aircraft Designs. *International Journal of Environmental Science and Development*, 4(2):187–190, April 2013.

[5] Frank H. Heppner. Avian Flight Formations. *Bird-Banding*, 45(2):pp. 160–169, Spring 1974.

[6] John P. Badgerow. An Analysis of Function in the Formation Flight of Canada Geese. *The Auk*, 105(4):pp. 749–755, 1988.

[7] P. B. S. Lissaman and Carl A. Shollenberger. Formation Flight of Birds. *Science*, 168(3934):1003–1005, 1970.

[8] D. Hummel. Aerodynamic Aspects of Formation Flight in Birds. *Journal of Theoretical Biology*, 104(3):321 – 347, 1983.

[9] H. Weimerskirch, J. Martin, Y. Clerquin, P. Alexandre, and S. Jiraskova. Energy Saving in Flight Formation. *Nature*, 413(6857):697–698, 2001.

[10] C. Cutts and J. Speakman. Energy Savings in Formation Flight of Pink-Footed Geese. *Journal of Experimental Biology*, 189(1):251–261, 1994.

[11] W.B. Blake and D.R. Gingras. Comparison of Predicted and Measured Formation Flight Interference Effects. *AIAA Paper*, (AIAA- 2001-4136), 2001.

[12] Kevin R. Walsh M. Jake Vachon, Ronald J. Ray and Kimberly Ennix. F/A-18 Performance Benefits Measured During the Autonomous Formation Flight Project. Technical Memorandum NASA/TM-2003-210734, NASA Dryden Flight Research Center, Edwards, California, September 2003.

[13] S.A. Ning, T.C. Flanzer, and I.M. Kroo. Aerodynamic Performance of Extended Formation Flight. *Journal of aircraft*, 48(3):855–865, 2011.

[14] J Scott. "Regions of upwash and downwash created by trailing vortices", diagram. Aerospaceweb, 17 July 2005, http://www.aerospaceweb.org/question/nature/q0237.shtml.

[15] John D. Anderson. *Fundamentals of Aerodynamics*. Number ISBN 0-07-237335-0 in McGraw-Hill Higher Education - Series in Aeronautical and Aerospace Engineering. Thomas E. Casson, New York, third edition, 2001.

[16] M.M. Munk. The Minimum Induced Drag of Airfoils. *Classical Aerodynamic Theory*, 1050:95, 1979.

[17] W. Blake and D. Multhopp. Design, performance and modeling considerations for close formation flight. In *AIAA Guidance, Navigation and Control Conference*, 1998.

[18] D. Hummel. The Use of Aircraft Wakes to Achieve Power Reductions in Formation Flight. In *AGARD FDP Symposium on The Characterization & Modification of Wakes from Lifting Vehicles in Fluid"*. DTIC Document, 1996.

[19] R.J. Ray, B.R. Cobleigh, M.J. Vachon, and C. St. John. Flight Test Techniques Used to Evaluate Performance Benefits During Formation Flight. Technical Report NASA/TP-2002-210730, NASA Dryden Flight Research Center, Edwards, California, August 2002.

[20] Michael J. Allen Curtis E. Hanson, Jack Ryan and Steven R. Jacobson. An Overview of Flight Test Results for a Formation Flight Autopilot. Technical Memorandum NASA/TM-2002-210729, NASA Dryden Flight Research Center, Edwards, California, August 2002.

[21] G. Ribichini and E. Frazzoli. Efficient coordination of multiple-aircraft systems. In *42nd IEEE Conference on Decision and Control*, volume 1, pages 1035–1040. IEEE, 2003.

[22] G.C. Bower, TC Flanzer, and I.M. Kroo. Formation Geometries and Route Optimization for Commercial Formation Flight. In *AIAA Applied Aerodynamics Conference*, number AIAA-2009-3615, 2009.

[23] T. E. Kent and A. G. Richards. A Geometric Approach to Optimal Routing for Commercial Formation Flight. In *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, Minnesota, August 2012.

[24] James Kless, Michael J. Aftosmis, S. Andrew Ning, , and Marian Nemec. Inviscid Analysis of Extended Formation Flight. In *Seventh International Conference on Computational Fluid Dynamics*, number D7-2012-4306, Big Island, Hawaii, July 2012. Science and Technology Corporation, NASA Ames Research Center, Stanford University, Stanford.

[25] Nicholas Bizinos and Christiaan Redelinghuys. Tentative Study of Passenger Comfort During Formation Flight Within Atmospheric Turbulence. *Journal of Aircraft*, 50(3):886–900, 2013.

[26] D. C. Burnham and J. N. Hallock. Chicago monostatic acoustic vortex sensor system. Technical Report DOT-TSC-FAA-79-103.IV, U.S. Department of Transportation, July 1982.

[27] N. Bizinos. Passenger Comfort During Formation Flight within Atmospheric Turbulence. Master's thesis, University of Cape Town, 2012.

[28] M.V. Cook. *Flight Dynamics Principles*. Elsevier Ltd., second edition, 2007.

[29] Stacey Gage. Creating a unified graphical wind turbulence model from multiple specifications. In *AIAA Modelling and Simulation Technologies Conference and Exhibit*, number AIAA-2003-5529, Natick, Massachusetts, August 2003. The Mathworks Inc.

[30] Military Specification: Flying Qualities of Piloted Airplanes, November 1980.

[31] Flying Qualities of Piloted Aircraft, December 1997.

[32] C. J. Houbolt, R. Steiner, and K. G. Pratt. Dynamic response of airplanes to atmospheric turbulence including flight data on input and response. Technical report, NASA Langley Research Center, 1964.

[33] Ronald D Flack. *Fundamentals of Jet Propulsion with Applications*. Number ISBN-13 978-0-521-81983-1. Cambridge University Press, New York, 2005.

[34] Ahmed F El-Sayed. *Aircraft Propulsion and Gas Turbine Engines*. CRC Press Boca Raton (Florida), 2008.

[35] Vivek Sanghi, B. K. Lakshmanan, and V . Sundararajan. Digital Simulator for Steady-State Performance Prediction of Military Turbofan Engine. *Journal of Propulsion and Power*, 14(1):74 – 81, January - February 1998.

[36] John S. McKinney. Simulation of a Turbofan Engine: Part 1. Description of Method and Balancing Technique. Technical Report AFAPL-TR-67-125, USAF, Air Force Aero Propulsion Laboratory, Wright-Patterson Air Force Base, Ohio, November 1967.

[37] John S. McKinney. Simulation of a Turbofan Engine: Part 2. User Manual and Computer Program Listing. Technical Report AFAPL-TR-67-125, USAF, Air Force Aero Propulsion Laboratory, Wright-Patterson Air Force Base, Ohio, November 1967.

[38] HIH Saravanamuttoo and BD MacISAAC. An Overview of Engine Dynamic Response and Mathematical Modeling Concepts. In *AGARD Conference Proceedings*, number 324, pages 25–1. AGARD, 1983.

[39] Ceri Evans. Testing and Modelling Aircraft Gas Turbines: An Introduction and Overview. In *UKACC International Conference on CONTRO*, number 455. University of Glamorgan, Wales, UK, September 1998.

[40] Vivek Sanghi, BK Lakshmanan, and V Sundararajan. Survey of advancements in jet-engine thermodynamic simulation. *Journal of Propulsion and Power*, 16(5):797–807, 2000.

[41] Melvin E. LaVerne and Aaron S. Boksenbom. Frequency-response of linear systems from transient data. Technical Report 977, National Advisory Committee for Aeronautics, April 1949.

[42] Aaron S. Boksenbom and Richard Hood. General algebraic method applied to control analysis of complex engine types. Technical Report 980, National Advisory for Aeronautics, April 1949.

[43] Edward W Otto and Burt L. Taylor III. Dynamics of a turbojet engine considered as a quasi-static system. Technical Note 2091, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, May 1950.

[44] Burt L. Taylor III and Frank L. Oppenheimer. Investigation of frequency-response characteristics of engine speed for a typical turbine-propeller engine. Technical Report 1017, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, March 1950.

[45] Seymour C. Himmel and Richard P. Krebs. The effect of changes in altitude on the controlled behavior of a gas-turbine engine. *Journal of the Aeronautical Sciences (Institute of the Aeronautical Sciences)*, 18(7):433–441, July 1951.

[46] Marcel Dandois and David Novik. Application of linear analysis to an experimental investigation of a turbojet engine with proportional speed control. Technical Note 2642, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, February 1952.

[47] Herbert Heppler, David Novik, and Marcel Dandois. Some dynamic characteristics of a turbojet engine for large accelerations. Research Memorandum E52H04, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, August 1952.

[48] JR Ketchum and RT Craig. Simulation of linearized dynamics of gas-turbine engines. Technical Note 2826, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, November 1952.

[49] G. J. Pack and Jr W. E. Phillips. Analog study of interacting and non-interacting multiple-loop control systems for turbojet engines. Technical Report 1212, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, 1954.

[50] David Novik and Herbert Heppler. Experimental determination of linear dynamics of two-spool turbojet engines. Research Memorandum E54J11, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, 1954.

[51] H. I. H. Saravanamuttoo and A. J. Fawke. Digital computer methods for the prediction of gas turbine dynamic response. *Society of Automotive Engineers*, pages 1805–1813, 1971.

[52] James F Dugan. Two-spool matching procedures and equilibrium characteristics of a two-spool turbojet engine. Research Memorandum E54J11, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, June 1954.

[53] James F Dugan. Component operating trends during acceleration and deceleration of two hypothetical two-spool turbojet engines. Research Memorandum E54L28, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, April 1955.

[54] Richard E. Filippi and James F Dugan. Effect of design over-all compressor pressure ratio division on acceleration characteristics of

three hypothetical two-spool turbojet engines. Research Memorandum E56D13, Lewis Flight Propulsion Laboratory, National Advisory Committee for Aeronautics, August 1956.

[55] V. L. Larrowe, M. M. Spencer, and M. Tribus. A dynamic performance computer for gas turbine engines. IP 206, The University of Michigan, Industry Program of the College of Engineering, January 1957.

[56] V. L. Larrowe, M. M. Spencer, and S. R. Lampert. An application of a gas turbine engine performance computer to a twin-spool engine. IP 250, The University of Michigan, Industry Program of the College of Engineering, November 1957.

[57] V. L. Larrowe and M. M. Spencer. Analog computer simulation of gas turbine engines for control study. *Society of Automotive Engineers*, 9, 1965.

[58] H. I. H. Saravanamuttoo. Analogue computer study of the transient performance of the orenda 600 h.p. regenerative gas turbine. *American Society of Mechanical Engineers*, (Paper 63, AHGT-38), 1963.

[59] H. I. H. Saravanamuttoo and A. J. Fawke. Simulation of gas turbine dynamic performance. *American Society of Mechanical Engineers*, (Paper 70, GT-23), 1970.

[60] A. J. Fawke, H. I. H. Saravanamuttoo, and M. Holmes. Experimental Verification of a Digital Computer Simulation Method for Predicting Gas Turbine Dynamic Behaviour. In J. C. Mundy, editor, *The Institution of Mechanical Engineers*, volume 186 27/72, pages 323 – 329, August 1972.

[61] James F. Sellers and Carl J. Daniele. DYNGEN - A Program for Calculating Steady-State and Transient Performance of Turbojet and Turbofan Engines. Technical Note D-7901, Lewis Research Center, NASA, Cleveland, Ohio, April 1975.

[62] Robert W. Koenig and Laurence H. Fishbach. GENENG - A Program for Calculating Design and Off-Design Performance of Turbojet and Turbofan Engines. Technical Memorandum D-6552, NASA, Lewis Research Center, 1972.

[63] Laurence H. Fishbach and Robert W. Koenig. GENENG ii - A Program for Calculating Design and Off-Design Performance of Two- and Three-Spool Turbofans with as Many as Three Nozzles. Technical Memorandum D-6553, NASA, Lewis Research Center, Cleveland, Ohio, 1972.

[64] Carl J. Daniele, Susan M. Krosel, and John R. Szuch. Digital Computer Program for Generating Dynamic Turbofan Engine Models (DIGTEM). Technical Memorandum 83446, NASA Lewis Research Center, Cleveland, Ohio, September 1983.

[65] John R Szuch. Hydes: A generalized hybrid computer program for studying turbojet or turbofan engine dynamics. Technical Memorandum X-3014, NASA Lewis Research Center, Cleveland, Ohio, 1974.

[66] Gerald G. Sadler and Kevin J. Melcher. DEAN: A Program for Dynamic Engine Analysis. Technical Memorandum 87033, NASA, Lewis Research Center, Cleveland, Ohio, July 1985.

[67] Naveed ur Rahman. *Propulsion and Flight Controls Integration for the Blended Wing Body Aircraft*. PhD thesis, Cranfield University, 2009.

[68] John B. Heywood. *Internal Combustion Engine Fundamentals*. Number ISBN 0-07-100499-8 in Automotive Technology Series. McGraw-Hill International, 1998.

[69] WPJ Visser. Gas turbine simulation at nlr. In *Making it REAL, CEAS Symposium on Simulation Technology (paper MOD05), Delft, the Netherlands*, 1995.

[70] J Yin. High bypass ratio fan modelling (wp2). *Performance Engineering UTC Annual Review Report*, 1999.

[71] J Yin. Proposed calculation procedures for the generation and application of 2-d fan characteristics. *Performance Engineering UTC Document PE005. Version 2*, 1999.

[72] B. Curnock, J. Yin, R. Hales, and P. Pilidis. High-bypass turbofan model using a fan radial-profile performance map. *Elsevier Science Ltd.Aircraft Design*, 4:115 – 126, 2001.

[73] J. Yin and P. Pilidis. Influence of Inlet Profile on High-BPR Turbofan Performance using a Radial Profile Map. In *ICAS*, 2002.

[74] Man San Li, Jun Fei Yin, and Barry Curnock. Low Bypass Ratio Turbofan Performance Modeling with Fan Radial Flow Profiles. In *Proceedings of the International Gas Turbine Congress*, number IGTC2003T okyo TS-094, Tokyo, November 2003.

[75] F. C. Mund and P. Pilidis. Performance simulation of a high-bypass turbofan with a 2D representation of the intake and fan components. *The Aeronautical Journal*, 112(1137):673 – 682, August 2008.

[76] Georgios Doulgeris, Hossein Khaleghi, Anestis Kalfas, and Pericles Pilidis. Development of a method for enhanced fan representation in gas turbine modeling. *International Journal of Rotating Machinary*, 2011(ID 182906), 2011.

[77] A. Alexiou and K. Mathioudakis. Development of a Turbofan Performance Model Using a Generic Simulation Tool. In *ASME Turbo Expo 2005: Power for Land, Sea and Air*, pages 1 – 9, Reno-Tahoe, Nevada, US, June 6-9 2005.

[78] J. Kurzke. Transient Simulations During Preliminary Conceptual Engine Design. *American Institute of Aeronautics and Astronautics*, (ISABE-2011-1321), 2011.

[79] Joachim Kurzke. Advanced User-Friendly Gas Turbine Performance Calculations on a Personal Computer. In *International Gas Turbine and Aeroengine Congress and Exposition*, number 95-GT-147, Housten, Texis, June 5-8 1995. ASME.

[80] Annette E. Nielsen, Christopher W. Moll, and Stephan Staudacher. Modeling and Validation of the Thermal Effects on Gas Turbine Transients. *Journal of Engineering for Gas TUrbines and Power*, 127:564 – 572, July 2005.

[81] Cleverson Bringhenti and Joo Roberto Barbosa. Effects of Turbine Tip Clearance on Gas Turbine Perfromance. In *Proceedingsof ASME Turbo Expo 2008: Power for Land, Sea and Air*, number GT2008-50196, Berlin, Germany, June 9-13 2008.

[82] Joachim Kurzke. How to get compnent maps for aircraft gas turbine performance calculations. In *International Gas Turbine and Aeroengine Congress and Exhibition*, number ASME 96-GT-164, pages 1 – 7, Bermingham, UK, June 10-13 1996.

[83] T. Schobeiri, M. Attia, and C. Lippke. GET RAN: A Generic, Modularly Structured Computer Code for Simulation of Dynamic Behavior of Aero-and Power Generation Gas Turbine Engines. *American Society of Mechanical Engineers, Journal of Engineering for Gas Turbines and Power*, 116:483–483, 1994.

[84] RW Claus, AL Evans, JK Lylte, and LD Nichols. Numerical propulsion system simulation. *Computing Systems in Engineering*, 2(4):357–364, 1991.

[85] Russell W Claus, Austin L Evans, and Gregory J Follen. Multidisciplinary propulsion simulation using npss. *AIAA Paper*, 92:4709, 1992.

[86] Colin K. Drummond, Gregory J. Follen, and Charles W. Putt. Gas Turbine System Simulation: An Object-Oriented Approach. Technical Memorandum 106044, NASA Lewis Research Center, Cleveland, Ohio, April 1992.

[87] Patrick T Homer and Richard D Schlichting. Support Heterogeneity and Distribution in the Numerical Propulsion System Simulation Project. In *High Performance Distributed Computing, 1993., Proceedings the 2nd International Symposium on*, pages 187–195. IEEE, 1993.

[88] Patrick T Homer and Richard D Schlichting. Using Schooner to Support Distribution and Heterogeneity in the Numerical Propulsion System Simulation Project. *Concurrency: Practice and Experience*, 6(4):271–287, 1994.

[89] Brian P. Curlett and James L. Felder. Object-Oriented Approach for Gas Turbine Engine Simulation. Technical Memorandum 106970, NASA Lewis Research Center, Cleveland, Ohio, July 1995.

[90] John A Reed, Abdollah A Afjeh, Patrick T Homer, Henry Lewandowski, and Richard D Schlichting. Development of an intelligent monitoring and control system for a heterogeneous numerical propulsion system simulation. In *Simulation Symposium, 1995., Proceedings of the 28th Annual*, pages 278–287. IEEE, 1995.

[91] John A Reed and Abdollah A Afjeh. An object-oriented framework for distributed computational simulation of aerospace propulsion systems. In *COOTS*, pages 149–164, 1998.

[92] John A Reed and Abdollah A Afjeh. Comutational Simulation of Gas Turbines: Part 1 - Foundations of Component-Based Models. In *International Gas Turbine & Aeroengine Congress & Exhibition*, number 99-GT-346, Indianapolis, Indiana, June 1999. ASME.

[93] John A Reed and Abdollah A Afjeh. Comutational Simulation of Gas Turbines: Part 2 - Extensible Domain Framework. In *International Gas Turbine & Aeroengine Congress & Exhibition*, number 99-GT-347, Indianapolis, Indiana, June 1999. ASME.

[94] John K Lytle, David A Remaklus, and Lester D. Nichols. Numerical Propulsion System Simulation. In *The Third Air Force/NASA Symposium on Recent Advances in Multidisciplinary Analysis and Optimization*, number 94N-71431, pages 115 – 121, Cleveland, Ohio, January 1999. NASA Lewis Research Center.

[95] John K. Lytle. The Numerical Propulsion System Simulation: An Overview. Technical Memorandum 2000-209915, NASA Glenn Research Center, Cleveland, Ohio, February 2000.

[96] Gregory J. Follen. An Object Oriented Extensible Architecture for Affordable Aerospace Propulsion Systems. In *RTO A VT Symposium on "Reduction of Military Vehicle Acquisition Time and Cost through Advanced Modelling and Virtual Simulation"*, 2002.

[97] Russell W. Claus, Thomas Lavelle, Scott Townsend, and Mark Turner. Coupled high-fidelity engine system simulation. In *26th International Congress of the Aeronautical Sciences*, 2008.

[98] Russell W. Claus and Scott Townsend. A Review of High Fidelity, Gas Turbine Engine Simulations. In *27th International Congress of the Aeronautical Sciences*, 2010.

[99] Khary I. Parker and Ten-Heui Guo. Development of a turbofan engine simulation in a graphical simulation environment. Technical Memorandum NASA/TM2003-212543, Glenn Research Center, Cleveland, Ohio, August 2003.

[100] Khary I. Parker and Kevin J. Melcher. The Modular Aero-Propulsion System Simulation (MAPSS) User's Guide. Technical Memorandum NASA/TM2004-212968, NASA, Glenn Research Center, Cleveland, Ohio, March 2004.

[101] Dean K. Frederick. User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS). Technical Memorandum NASA/TM2007-215026, NASA Glenn Research Center, Cleveland, Ohio, October 2007.

[102] Jonathan A. DeCastro, Jonathan S. Litt, and Dean K. Frederick. A Modular Aero-Propulsion System Simulation of a Large Commercial Aircraft Engine. Technical Memorandum TM2008-215303, NASA Glenn Research Center, Cleveland, Ohio, July 2008.

[103] Ryan D. May, Jeffrey Csank, Thomas M. Lavelle, Jonathan S. Litt, and Ten-Huei Guo. A High-Fidelity Simulation of a Generic Commercial Aircraft Engine and Controller. Technical Memorandum TM2010-216810, NASA Glenn Research Center, Cleveland, Ohio, July 2010.

[104] J. Kurzke. *Smooth C 8.2 - Preparing Compressor Maps for Gas Turbine Performance Modeling.* GasTurb, Dachau, Germany, 2009.

[105] J. Kurzke. *GasTurb 11 - Design and Off-Design Performance of Gas Turbines.* GasTurb, Dachau, Germany, 2009.

[106] A Stamatis, K Mathioudakis, J Ruiz, and B Curnock. Real time engine model implementation for adaptive control & performance monitoring of large civil turbofans. *ASME paper*, (2001-GT):0362, 2001.

[107] A. Alexiou and K. Mathioudakis. Gas Turbine Engine Performance Model Application Using an Object-Oriented Simulation Tool. In *ASME Turbo Expo 2006: Power for Land, Sea and Air*, number GT2006-90339, Barcelona, Spain, May 2006.

[108] A. Alexiou, E.H. Baalbergen, O. Kogenhop, K. Mathioudakis, and P. Arendsen. Advanced Capabilities for Gas Turbine Engine Performance Simulations. In *ASME Turbo Expo 2007: Power for Land, Sea and Air*, number NLR-TP-2007-513, Montreal (Canada), May 2007. NLR and National Technical University of Athens (Greece).

[109] W.P.J. Visser and S.C.A. Kluiters. Modelling the effects of operating conditions and alternative fuels on gas turbine performance and emissions. In *RTO-AVT "Gas Turbine Engine Combustion, Emissions and Alternative Fuels" symposium*, Lisbon, Portugal, October 1998.

[110] W. P. J. Visser and M. J. Broomhead. Gsp a generic object-oriented gas turbine simulation environment. In *ASME Turbo Expo*, Munich, Germany, May 2000.

[111] W.P.J. Visser, O. Kogenhop, and M. Oostveen. A generic approach for gas turbine adaptive modeling. In *ASME Turbo Expo*, Vienna, Austria, June 2004.

[112] C William Gear. *Numerical initial value problems in ordinary differential equations*. Prentice Hall PTR, 1971.

[113] Oscar Kogenhop, Michael J. Broomhead, and Wilfried P.J. Visser. *Gas Turbine Simulation Program GSP 11 User Manual*. National Aerospace Laboratory NLR, Amsterdam, Amsterdam, gsp 11.1.3.0 edition, November 2011.

[114] Boeing. Boeing 747-100 Characteristics. Accessed 05 December 2013, (http://www.boeing.com/boeing/commercial/startup/747100.page).

[115] Robert K. Heffley and Wayne F. Jewell. Aircraft Handling Qualities Data. Contractor Report NASA CR-2144, Systems Technology Inc, Hawthorne, California 90250, December 1972.

[116] M. W. Chase. Nist-janaf thermodynamical tables. *Journal of Physical and Chemical Reference Data Monographs & Suppliments*, 4(Monograph No 9), 1998.

[117] Bonnie J. McBride, Michael J. Zehe, and Stanford Gordon. NASA Glenn Coefficients for Calculating Thermodynamic Properties of Individual Species. Technical Paper NASA TP-2002-211556, NASA Clenn Research Center, Cleveland, Ohio, September 2002.

[118] J. Seddon and E.L. Goldsmith. *Intake aerodynamics*. Amer Inst of Aeronautics &, 1999.

[119] Chris Drummond and Craig R. Davison. Improved Ccompressor Maps using Appropriate Solutions to the Moore Greitzer Model. In *Proceedings of ASME Turbo Expo: Power for Land, Sea and Air*, number GT2009-60148, Orlando, Florida, USA, June 2009. National Research Council of Canada.

[120] F.K. Moore and E. M. Greitzer. A Theory of Post-stall Transients in Multistage Axial Compression Systems. Contractor Report 3878, NASA Lewis Research Center, 1985.

[121] Andrea Lazzaretto and Andrea Toffolo. Analytical and Neural Network Models for Gas Turbine Design and Off-Design Simulation. *Applied Thermodynamics*, 4(4):173 – 182, December 2001.

[122] G. L. Converse and R. G. Griffin. Extended Parametric Representation of Compressor Fans and Turbines: Volumes i - iii. Contractor Report NASA-CR-174645, NASA Lewis Research Center, 1984.

[123] K. Ghorbanian and M. Gholamrezaei. An Artificial Neural Network Approach to Compressor Performance Prediction. *Applied Energy*, 86:1210 – 1221, 2009.

[124] Youhong Yu, Lingen Chen, Fengrui Sun, and Chih Wu. Neural-Network Based Analysis and Prediction of a Compressor's Characteristic Performance Map. *Applied Energy*, 84:48 – 55, 2007.

[125] Chris Drummond and Craig R. Davison. Capturing the Shape Variance in Gas Turbine Compressor Maps. In *Proceedings of ASME Turbo Expo: Gas Turbine Technical Congress and Exposition*, number GT2009-60141, Orlando, Florida, USA, 2009. National Research Council of Canada.

[126] J. Kurzke and C. Riegler. A new compressor map scaling procedure for preliminary conceptional design of gas turbines. *ASME Turbo Expo, Munich, Germany*, 2000.

[127] J. Kurzke. *Smooth T 8.2 - Preparing Turbine Maps for Gas Turbine Performance Modeling*. GasTurb, Dachau, Germany, 2009.

[128] H. I. H. Saravanamuttoo, G. F. C. Rogers, H. Cohen, and P. V. Straznicky. *Gas Turbine Theory*. Number ISBN: 978-0-13-222-437-6. Pearson Education Limited, 6 edition, 2009.

[129] ISO. Standard Atmosphere. Technical Report ISO 2533:1975, ISO, Last Reviewed 2007.

[130] S.R. Turns. *An Introduction to Combustion: Concepts and Applications.* McGraw-hill New York, second edition, 1996.