

21 of 13
21/1/17

11/1/17

Development and Validation of a Computer Assisted Design Methodology for Gas- Turbine-Based Aircraft Engines

By: Kamran Eftekhari Shahroudi

Development and Validation of a Computer Assisted Design Methodology for Gas-Turbine-Based Aircraft Engines

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. ir. K. F. Wakker,
in het openbaar te verdedigen ten overstaan van een commissie,
door het College van Dekanen aangewezen,
op vrijdag 16 december 1994 te 10.30 uur
door

Kamran EFTEKHARI SHAHROUDI

**Master of Science in Engineering
(Aerospace Engineering)
University of Michigan, Ann Arbor, USA
Geboren te Tehran, Iran**

Dit proefschrift is goedgekeurd door de promotoren:

Prof.ir. E. Torenbeek

Prof.ir. J.P. van Buijtenen

Published and distributed by:

Delft University Press

Stevinweg 1

2628 CN Delft

The Netherlands

Telephone +31 15 783254

Fax +31 15 781661

CIP-DATA KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Eftekhari Shahroudi, K.

Development and Validation of a Computer Assisted Design Methodology for
Gasturbine-Based Aircraft Engines / K. Eftekhari Shahroudi. - Delft : Delft
University Press. - Ill.

Thesis Delft University of Technology. - With ref. - With summary in Dutch.

ISBN 90-407-1070-8

NUGI 832

Subject headings: CAD, Aircraft engines

Copyright © 1994 by K. Eftekhari Shahroudi

All rights reserved.

No part of the material protected by this copyright notice may be reproduced
or utilized in any form or by any means, electronic or mechanical, including
photocopying, recording or by any information storage and retrieval system,
without permission from the publisher: Delft University Press, Stevinweg 1,
2628 CN Delft, The Netherlands.

Printed in The Netherlands

*"This thesis is dedicated to Ms.
Roghieh Sokhan Bakhsh and
Mr. Manouchehr Eftekhari
Shahrودي, my beloved
parents, who sacrificed
everything to provide me with
the opportunity to realize my
dreams."*

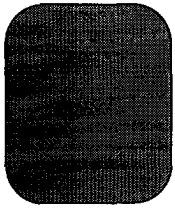


Table Of Contents

Table Of Contents .1

Abstract .7

Nomenclature, symbols and abbreviations .11

0.1 General Terms, Acronyms and Abbreviations .11

0.2 Glossary for Aerothermodynamic Design Approach .12

0.3 Glossary For The Adaptive Grid & Multivariate Newton Raphson .16

1 Introduction 1

1.1 The Natural Design Cycle (NDC) 3

1.2 Motivation Behind Developing CAGED 11

1.2.1 Integrated Engine and Aircraft Approach 11

1.2.2 Need for an Independent "Stand Alone" System 15

1.2.3 Need for General Approach to Engine Configuration and

	<i>Fast Engine Models</i>	16
	1.2.4 <i>Need for Real Time Optimization Methods</i>	17
1.3	<i>SCOPE of the CAGED engine models</i>	18
1.4	<i>System Requirements Summary</i>	19
2	<i>Basic Approaches and Tools for obtaining “Real Time” and “Natural Design Cycle”</i>	21
2.1	<i>Conceptual Changes</i>	24
	2.1.1 <i>Engine Modelling</i>	24
	2.1.2 <i>High Speed Function Generation from Data</i>	27
	2.1.3 <i>High Speed Optimization and Solution to n simultaneous nonlinear equations</i>	28
	2.1.4 <i>High Speed manipulation of accurate data from other sources than CAGED</i>	29
	2.1.5 <i>Fluid Properties from Chemical Equilibrium Calculations</i>	29
	2.1.6 <i>Non-iterative Off- Design Methods</i>	31
2.2	<i>Real Time Movement Through Design Space</i>	34
	2.2.1 <i>Dynamic Histograms “DynHist”</i>	35
	2.2.2 <i>Dynamic Carpets “DynCarp”</i>	36
2.3	<i>Graphical User Interface GUI</i>	38
3	<i>Brief Review of Past and Current Activities in Computer Aided Engine Design / Analysis</i>	39
4	<i>Aerothermodynamic Design Approach</i>	43
4.1	<i>Engine Configuration Definition</i>	44
	4.1.1 <i>The Universal Component</i>	44

-
-
- 4.1.2 Interaction between Components 47**
 - 4.2 On Design Method 48**
 - 4.2.1 Source Code Generation for On-Design Performance of Engine Models & "Serial Network Logic" 48**
 - 4.3 Cycle Selection and Sizing 54**
 - 4.4 Off-Design Analysis 54**
 - 4.4.1 Direct Non-Iterative Solution For a Generic Spool 57**
 - 4.4.1.1 Spool With Restricted Outlet 60**
 - 4.4.1.2 Burner or Duct Pressure Loss Term 65**
 - 4.4.1.3 Pressure Gradient Term of Downstream Components 67**
 - 4.4.1.4 Generic Spool With Non-Restrictive Expanding Downstream Components 69**
 - 4.4.1.5 The Intake Recovery Temperature Ratio Term 72**
 - 4.4.1.6 Generic Spool with Well Designed and Well Matched Components 74**
 - 4.4.1.7 The Mass Ratio Term from Bypass Nozzle Characteristics 79**
 - 4.4.1.8 The Mass Ratio Term from Electrical Analogy 81**
 - 4.4.1.9 Correction for Power Off-Take and Variation of Compressor Efficiency 83**
 - 4.4.1.10 Inclusion of a Generic Spool within Another Generic Spool 84**
 - 4.4.2 Summary of Direct Analytical Off-Design Equations in Closed Form 85**
 - 4.4.3 Strategy for using the Non-Iterative Off-Design Equations and Source Code Generation for Off Design 87**
 - 4.5 Transient Behaviour of a Generic Spool 91**
 - 4.6 Significance of High Speed On and Off-Design Engine Models 94**
 - 4.7 Earlier Work on Application of Differential Analysis to Engine Off-design Equations 95**

5 *Integrated Thermodynamic and Mechanical Design Analysis Approach 99*

5.1 *Engine Geometry Determined by Engine Thermodynamics 100*

5.2 *Engine Thermodynamics Determined by Engine Geometry 108*

5.3 *Hybrid Integrated Thermodynamic and Mechanical Design 111*

5.3.1 *Sensitivity formulae via analysis 112*

5.3.2 *Sensitivity formulae from data base 115*

6 *Numerical Optimization and Solution Methods 117*

6.1 *Non-Random Adaptive Grid Method for Fast Optimization of Highly Dimensional, Badly Behaving Real Time Functions 119*

6.1.1 *Some Fundamental Distinctions 120*

6.1.2 *Search Strategies 123*

6.1.3 *The Non-Adaptive Random Search 125*

6.1.4 *The Non-Random Adaptive Grid Search 125*

6.1.4.1 *The Basic Search Strategy 125*

6.1.4.2 *The Coupling Relations for the Non-Random Adaptive Grid Method 126*

6.1.5 *Grid Adaptation Using Imaginary Flexible Beam 133*

6.1.6 *Practical Issues 136*

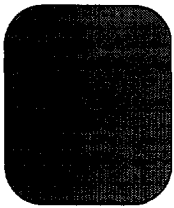
6.1.7 *The Effect Of Randomness On Adaptation (Break-Even Point) 141*

6.2 *Variations of the Basic Theme 145*

6.2.1 *Inclusion of Constraints In the Search Strategy 145*

-
-
- 6.2.2 Multiple optima 146**
 - 6.2.3 Independent Runs to Improve RoI 147**
 - 6.3 Results 148**
 - 6.4 Clarification of Logic Behind Derivations 149**
 - 6.4.1 Def2 and the Probability of Step Success 149**
 - 6.4.2 Behaviour of F and Constraints 150**
 - 6.4.3 Rigorous Accounting of the Type of Search Points and the Number of Affected Subregions 150**
 - 6.5 Summary for the Non-Random Adaptive Grid Search 151**
 - 6.6 The Modified Multivariate Newton Raphson 152**
 - 6.6.1 Multidimensional Function Generation From Data 153**
 - 6.6.2 Linear Solution to n Simultaneous Equations 158**
 - 7 Validation of Methods and Progress 161**
 - 7.1 Validation of on-design engine models 162**
 - 7.2 Validation of Off-design Engine Models 174**
 - 7.2.1 Modification to the original closed form equations 177**
 - 7.2.2 The High Pressure Spool Calculated with the Restricted Outlet Solution 178**
 - 7.2.3 The Low Pressure Spool Calculated with Unrestricted Outlet Solution 198**
 - 7.3 The Non-Random Adaptive Grid Numerical Optimization Technique 208**
 - 7.3.1 The Test Function 209**
 - 7.3.2 Criteria for Convergence 210**
 - 7.3.3 The Original Search Method 213**
 - 7.3.4 The Fast Alternative 215**
 - 7.3.5 The Original Scheme or the Fast Scheme? 219**

8	<i>Discussion and Conclusions</i>	221
8.1	<i>The Influence of Improving Computer Technology</i>	222
8.2	<i>Significance of this Research within the Context of Natural Design</i>	223
8.3	<i>Significance of this Research Out of the Context of NDC</i>	234
8.3.1	<i>The Graphic User Interface (GUI)</i>	235
8.3.2	<i>The non-random adaptive grid numerical optimization technique</i>	237
8.3.3	<i>The DynCarp and DynHist modules</i>	238
8.3.4	<i>Calculation of Steady State Off-Design Performance of Engines</i>	239
8.3.5	<i>Execution Speed of CAGED Engine Models</i>	241
8.4	<i>Future Work</i>	242
9	<i>References</i>	245
10	<i>Appendix</i>	257



Abstract

The *Computer Aided General Engine Design (CAGED)* doctorate research project was initiated in September, 1989 at the *Aircraft Design and Flight Mechanics Group*, Faculty of Aerospace Engineering, Delft University of Technology, The Netherlands.

The main objective of this research was to obtain a modern and efficient tool to assist engineers in the conceptual/preliminary phase of gas turbine based aircraft engine mechanical and thermodynamic design, by automating many of the laborious tasks involved while leaving all the intuitive and decision making tasks to the human designer.

CAGED aims to mimic the *Natural Design Philosophy* which is a *High Speed* design optimization loop with active short term *Involvement* of the *Human Designer*. In this way emphasis is made on enabling the human designer to gain new experiences and reach new conclusions quickly rather than aiding the design process by a collection of previously gained experiences and conclusions, i.e. not employing artificial intelligence concepts.

New methods and concepts were developed to enable *CAGED* to generate high speed engine models (e.g. order of milliseconds execution times on a 33 MHz SGI Iris Indigo). Among them,

1. The concept of *Universal Component*
2. *High Speed Direct* (i.e. non-iterative) *Analytical Solution* to *Off-Design* equations of a *Generic Spool*
3. *High Speed Direct* (i.e. non-iterative) method for transient behaviour of *Generic Spool*

4. Removal of *Sequential Network Logic* from the generated engine model
5. Including only variables of interest to current design in the model
6. Using high speed function generation for generation of fluid properties
7. Generate Engine Model in the form of uncompiled source code

These engine models are generated for any engine configuration and variables which are defined visually via the *GUI*. Their quick execution times and high accuracy makes them ideal for inclusion within large optimization programs (e.g. within an aircraft optimization routine).

Two new and efficient model exploration and optimization tools *DynCarp* (Dynamic Carpets) and *DynHist* (Dynamic Histograms) were developed. They enable ***Real Time Movement Through the Multi-dimensional Design Space*** when used in conjunction with high speed computational models. With these tools it is possible for the human designer to make hundreds of parameter variations a minute and grasp the meaning or consequences of these changes also within that time. This is particularly useful for studying a new concept or making a major change to an existing concept when the human user has no prior knowledge of the behaviour of this new concept. In addition ***High Speed Function Generation Routines*** were written which generate a high speed model of a data file with large number of independent variables, in order to make it possible to use *DynCarp* and *DynHist* with data files generated by other large and more accurate programs.

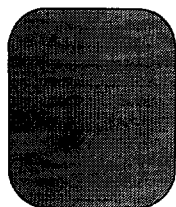
For cases where automatic numerical optimization is preferable (i.e. with the human designer out of the optimization loop), a new non-gradient guided numerical optimization routine based on the *Adaptive Grid* was developed, which is suitable for optimization of *High Speed Functions* (~milliseconds evaluation time) of not too high dimensionality (< 25). Further the *Coupling* (a quantitative relation ship) between required *Accuracy*, required *Probability* (that true optimum has been found) and *Calculation Effort* were analytically derived. This aids in determining the calculation effort, for a given required accuracy and probability function, ***Before*** the solution has started (i.e. a quantitative measure of the success of the optimization exercise, or *return on investment*, before commitment to start the search).

The *Multivariate Newton Raphson* numerical method is extensively used in engine analysis systems in general, due to its versatility and simplicity. The application of simple numerical tricks to this technique resulted in drastic improvement in execution speeds of this algorithm, making it more suitable for high speed function generation from data, solution of simultaneous equations and for optimization.

CAGED is intended for future integration with the *Aircraft Design Analysis System (ADAS)*, also developed at TU Delft, in order to make it possible to **include engine design variables within aircraft conceptual/preliminary design synthesis**. For practical reasons, rather than including the relatively large *CAGED* program as a subsystem of *ADAS*, the best solution was that only the engine models (i.e. source code) generated by *CAGED* would be included as a sub program of *ADAS* or other large synthesis programs.

A fully interactive *Graphical User Interface* was also developed for conceptual engine design. All system commands are *Visually Accessible* by using mouse and screen buttons. Any *Engine Configuration* can be represented by visually constructing a *Network of Standard Components* or combination of previously defined sub-models. This visual network can then be used for any future parameter get/set or modification operation.

For software performance and practical reasons, *CAGED* was developed to be an independent *Stand Alone System* with all software purpose developed and written by the author.



Nomenclature, symbols and abbreviations

For ease of use glossaries are subdivided into 3 groups,

1. General textual terms e.g. *CAGED*, *ADAS* etc.
2. Terms and Symbols used in Aerothermodynamic Analysis, Generic Spool Equations etc.
3. Terms and symbols used for numerical optimization methods. section 6 on page 117.

1 General Terms, Acronyms and Abbreviations

- ADAS :** Aircraft Design and Analysis System, a computer assisted aircraft preliminary design system also developed at *Delft TU* ^[3]
- CAD :** Computer Aided Design
- CAE :** Computer Aided Engineering
- CAGED :** Computer Aided General Engine Design System, a new conceptual/preliminary design package for gas-turbine based aircraft engines, implementing and validating the concept of *Natural Design Cycle*
- CEC :** Chemical Equilibrium Calculations
- CET86 :** Computer program for *CEC* ^[25]
- Delft TU :** Delft University of Technology
- DynCarp :** Dynamic Carpets, a computer program for human in the loop optimization of highly dimensional, high speed functions. Allows *Real Time Movement Through Design Space*. An independent module included in *CAGED* (See section 2.2.2 on page 36).

DynHist :	Dynamic Histograms, a computer program for human in the loop optimization of highly dimensional, high speed functions. Allows Real Time Movement Through Design Space . An independent module included in CAGED (See section 2.2.1).
EAR :	Externally Applied Relation
ERL :	Equilibrium Running Line is the off-design steady state operating line of a gas turbine spool
GASTURB :	Program for steady state calculation of various gas-turbine cycles ^[36]
GUI :	Graphical User Interface
HD :	Human Designer
NDC :	The Natural Design Cycle
NNEP :	or NNEPEQ or NEPAS : versions of the Navy Nasa Engine Program ^[16] from NASA LeRC
RoI :	Return on Investment for a numerical optimization technique
SNL :	Sequential Network Logic
w.r.t. :	With respect to

2 Glossary for Aerothermodynamic Design Approach

Note :

- Figure 22 on page 53, defines station numbers for the **Generic Spool** which are used as subscript, e.g. \dot{m}_I is the mass flow rate at inlet to compressor.
- Table 5 (in appendix) is a collection of analytical off-design equations by other authors, as they appear in the original texts and not using the notations of the present report.

A :	annulus or duct area
amb :	ambient
a_c :	polytropic exponent for compressor = $(\gamma\eta_c) / (\gamma - 1)$
a_t :	polytropic exponent for turbine = $\gamma / ((\gamma - 1) \eta_t)$
B :	constant defined in Table 6 (in appendix)

<i>byp</i> :	bypass flow
<i>C</i> :	constant defined in Table 6 (in appendix)
<i>C_p</i> :	specific heat at constant pressure
<i>C_{p_{rat}}</i> :	ratio C_{p_3}/C_{p_1}
<i>C_v</i> :	specific heat at constant volume
<i>E</i> :	constant from turbine velocity triangle
ERL :	equilibrium running line of spool
<i>e</i> :	shaft rotational energy + compressor work.
∇X :	non-dimensional gradient/derivative of scalar <i>X</i> w.r.t. μ_1 , see eq. 11
$\nabla_\mu X$:	non-dimensional gradient of scalar <i>X</i> w.r.t. mass flow parameter μ .
<i>F</i> :	constant from turbine velocity triangle
<i>G</i> :	scalar, see eq. 72
<i>I</i> :	electric current
<i>k</i> :	$1 - \frac{1}{\nabla_{\mu_{in}} \pi_n}$, see eq. 53
<i>M</i> :	Mach number
<i>M</i> :	mixer type of network component, Figure 16 on page 46
<i>m</i> :	mass flow rate
<i>m_{rat}</i> :	mass ratio or in-pass ratio, $m_3/m_1 = 1/(1 + \lambda)$
<i>N</i> :	normal type of network component, Figure 16 on page 46
<i>N_{Mix}</i> :	number of unknown or invisible mixers. See section 4.1.2 and Fig. 17
<i>N_{Run}</i> :	number runs of sequential network logic needed to solve network with <i>N_{Mix}</i>
<i>N_c</i> :	number of compressor stages
<i>N_t</i> :	number of turbine stages
<i>O</i> :	current transient operating point of spool
<i>O_{ERL}</i> :	a point on the <i>ERL</i> which has the closest energy level to point to <i>O</i> , used in transient analysis
<i>p</i> :	static pressure
<i>p_{amb}</i> :	ambient pressure
<i>p_t</i> :	stagnation pressure

Nomenclature, symbols and abbreviations

$P_{Off\ Take}$	shaft power Off-Take
q	dynamic pressure
R	gas constant (J/ Kg/K)
R_u	universal gas constant
S	splitter type of network component, Figure 16 on page 46
SNL	serial or Sequential Network Logic.
T	static temperature
T_{SL}	sea level ambient temperature
T_t	stagnation temperature
U	tangential speed of compressor or turbine at mean radius
V_Z	mean axial flow velocity at inlet to component
X_i	influence of variable i , tabulated in Table 6
Y_i	influence of variable i , tabulated in Table 6
Z	constant, see Table 6

Greek symbols

δ	stagnation pressure corrected to sea level ambient, p_t/p_{SL}
η	polytropic efficiency, subscript c or t , denotes compressor or turbine.
γ	ratio of specific heat capacities C_p/C_v
λ	bypass ratio = bypass mass flow / core mass flow
π	stagnation pressure ratio across component, $p_{t_{out}}/p_{t_{in}}$, subscript denotes which component
π_b	burner or inner-spool pressure ratio
π_c	compressor pressure ratio
π_n	nozzle pressure ratio, $p_{out}/p_{t_{in}}$
π_r	ram recovery pressure ratio, p_{t_1}/p_{amb}
π_s	generic spool pressure ratio, $= p_{t_4}/p_{t_1}$, see Fig. 22
π_t	turbine stagnation pressure ratio, <u>note</u> : out/in

Ψ :	Work done factor for turbomachine = $(C_p \Delta T_t) / U^2$
Ψ_t :	Work done factor for turbine
ϕ :	flow coefficient of first stage of turbomachine
ϕ_c :	flow coefficient of first compressor stage
ϕ_t :	flow coefficient of first turbine stage
μ :	mass flow parameter $(m \sqrt{RT_t}) / (Ap_t)$
μ_{in} :	mass flow parameter at inlet to component
μ_{out} :	mass flow parameter at outlet of component
ρ :	static density
ρ_t :	stagnation density
τ :	stagnation temperature ratio across component, $T_{t_{out}} / T_{t_{in}}$, subscripts denote component
τ_s :	generic spool stagnation temperature ratio, T_{t_4} / T_{t_1} in Figure 22 on page 53
\emptyset :	generic spool stagnation temperature ratio, T_{t_3} / T_{t_1} in Figure 22 on page 53
θ :	stagnation temperature corrected to sea level ambient, T_t / T_{SL}

Subscripts

byp :	bypass flow
SL :	sea level
c :	compressor
c_s :	compressor stage
b :	burner
d :	diffuser
f :	fan
i :	summation index
n :	nozzle
r :	intake recovery
s :	generic spool, see Fig. 22

t :	turbine
t_s :	turbine stage
$1-7$:	station numbers for the <i>Generic Spool</i> , see Figure 22 on page 53

3 Glossary For The Adaptive Grid & Multivariate Newton Raphson

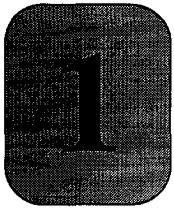
A :	gridded subspace of S
d :	dimensionality of F
$Def1$:	definition of step success used to estimate $Def2$
$Def2$:	robust definition of step success
E :	calculation Effort
\vec{E} :	error vector
F :	multidimensional merit function to be optimized
$f_1(x)$:	one dimensional function used to define the test function, see eq. 217
$f_2(y)$:	one dimensional function used to define the test function, see eq. 217
$f_3(z)$:	one dimensional function used to define the test function, see eq. 217
f :	shrinkage factor of A per step
I_K :	k^{th} independent variable
J :	step index
k :	dimension index
\vec{M} :	gradient matrix
N :	number of grid lines in each dimension
N_S :	number of function evaluation steps in optimization
$old \& new$:	grid positions before and after beam transformation
O :	optimum point of merit function F
O_1, O_2 etc.:	local optima other than O
P :	probability that the optimum has been found
P_{grid} :	contribution of gridding to probability of step success
P_{step} :	probability of a step being successful
S :	total solution space

s : sharpness factor, used in the definition of test function, see section 7.3.1 on page 209

SR : significant region of solution space S

Greek Symbols

ϵ : required inaccuracy or error



Introduction

The *Computer Aided General Engine Design (CAGED)* system was born in a predominantly aircraft design and flight mechanics group and not in an engine design group. The main motivation behind this project was to make it possible in future to include engine design parameters in the overall aircraft design analysis process. In practice this meant that *CAGED* had to somehow cooperate and interact with the already existing *Aircraft Design and Analysis System (ADAS)* ^[3] also developed at *Delft TU*.

Speaking strictly in the unclassified preliminary design domain, there exist several programs for propulsion analysis, in the thermodynamic and/or mechanical senses (See section 3). However it became clear at the outset of this project that no single design system existed that could satisfy the majority of our requirements. It was also not practical for performance, functionality and availability reasons to combine several of the existing programs into one large inefficient system.

The above reasoning combined with some stringent computing efficiency and functionality requirements from *CAGED*, forced the author to develop 100% of the software tailor made although it is much less effort to initially write some interfaces between several large and small systems. See section 1.2 for more detailed motivation behind various features of *CAGED*.

The development of *CAGED* involved developing several special features. These are not all specific to aircraft engine design but have application to *CAD / CAE* in general, despite the specific nature of program development. Namely;

- Fully Interactive Graphical User Interface
- General Engine Configurations

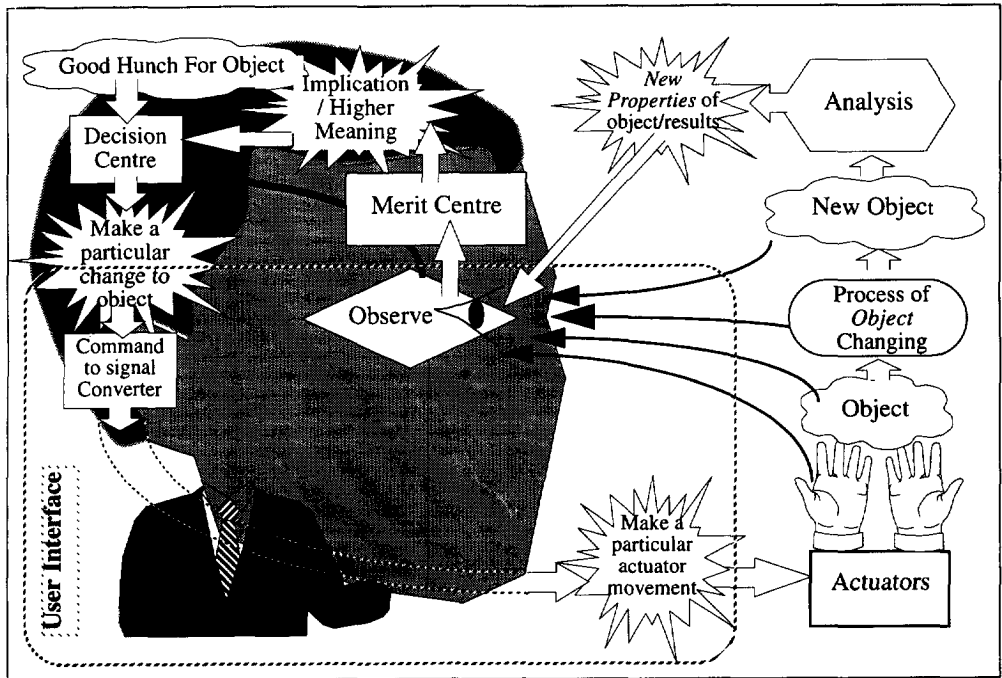


FIG. 1 Visualization of the *Design Cycle*. The design exercise is made up of continuous looping of the *Design Cycle* where the *Good Hunches* and the rules contained within the *Merit Centre* are continuously updatable.

- Generation of C source code for High Speed Engine Modelling
- Real Time Movement Through Design Space
- Non-iterative Thermodynamic Off-Design Methods
- Interactive mechanical design

See section 1.3 for more detailed description of these and other features of *CAGED*.

"*CAGED* is a design system which includes the human designer (*HD*) as its most significant module". While this sentence is perfectly clear to the author, it is possible to discuss the meaning of this last sentence for many hours. Regular discussions with other people interested in "design" has shown that different people understand different things by terms such as "*Design*", "*Analysis*", "*Inverse Design*", "*Real Time*", "*High*

Speed" etc. See section 1.1 for further clarification of what the author means by various terms which could be crucial to understanding the bulk of this work.

1.1 The Natural Design Cycle (NDC)

It is very instructive to have philosophical discussions about design when one works in a group with strong tradition in this field. Although these discussions hardly ever convince one person or the other or immediately lead to new and marvellous inventions, they illustrate very strongly that individuals even those influenced by working in the same group can have drastically different understanding by seemingly very simple terms such as "*Design*" and "*Analysis*".

This section attempts to describe very briefly what the author understands by the various terminology common to the field. This is done without any philosophical claim and just for clarification.

Further an attempt will be made to convince the reader that "*Real Time Optimization*" and "*High Speed Engine Modelling*" are not just some nice new gadgety terms but have real significance and application to the design process and offer very practical advantages to the propulsion engineer or to any designer/analyst who has difficulty understanding/designing a complex object with large number of influences.

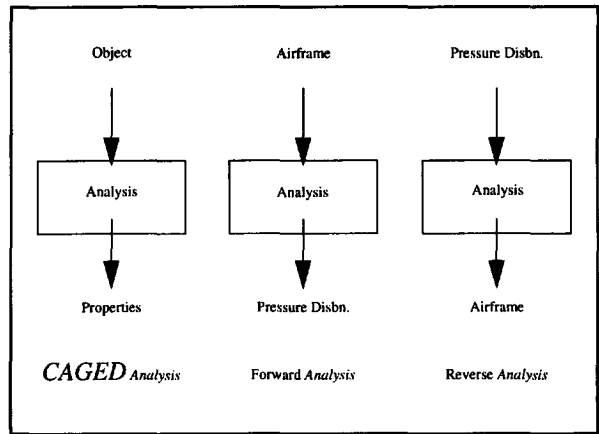
Fig. 1 shows the design cycle roughly as the author sees it. The activity starts with need for some objectⁱ. After this point the need and some vague description of the object exists. Here we do not attempt to describe how the need is generated and how the object is generated from this need. The object can be a specific physical or abstract object or a class of these objects. In the context of engine on-design thermodynamic analysis, the object will be the state of the engine model, i.e. the values of design variables.

Based on his previous experience, knowledge and ingenuity he generates a good hunch for the object. Generating good hunches is not a singular occurrence and the *HD* can be continuously generating theseⁱⁱ until design is terminated. *HD* then places this object in the "*New Object*" balloon in the figure. The "*New Object*" is then fed to the "*Analysis*" block. This block is responsible for finding the "*Properties*" that pertain to this *New Object*. In our example, the engine model source code would be the "*Analysis*" block

i. Object can be a physical or abstract object, e.g. bicycle, software, new social order, a method to make success out of marriage etc.

ii. If the Structure of The Overall System allow it.

FIG. 2 *CAGED Analysis* can represent both Forward and Reverse Analysis due to universal definitions of “Object” and “Properties”. E.g. both Airframe and Pressure Distribution can be considered as an object.



and dependant variables (e.g. specific fuel consumption, specific thrust etc.) would be the “*New Properties*”.

These “*New Properties*” and other observables are then fed to the “*Merit Centre*” via the “*Observe Centre*”. This block contains a set of rules for judging properties of objects. These rules come from experience and ingenuity of the human designer and are not necessarily fixed. i.e. can be continuously upgraded as design progresses. The “*Merit Centre*” generates implication or higher meaning of object properties. Higher Meaning is merit or deeper meaning than that contained in the “*New Properties*” of the “*Object*”. This is not to be confused with merit functions. Once it is possible to define a merit function in equation format, then it is *not* placed in the “*Merit Centre*”. It is simply placed in the “*Analysis*” block affecting the “*New Properties*”. In our example, “*Merit Centre*” answers the question: How good is the current combination of design parameters, all things considered.

Next in line is the “*Decision Centre*” which is responsible for deciding whether a change to the object is needed and in what particular way. The final decision made depends on the merit of the current object, new hunches and what the designer observes. For example the output of “*Decision Centre*” would be to increase fan pressure ratio by 1%. Any decision is the result of processing good hunches, merit and observation. The latter is not based on any human experience or memory and shows the current state of the model and its properties at the current point in the design space. Both merit and good hunches involve both long term and short term experience. Long term experience is knowledge/conclusions gained from previous runs of the design system or previous design/technical activity. Short term experience is that which is gained by successive loops of the high speed design cycle.

The designer is not constrained to observing only one particular point, i.e. the designer can be looking at the actuators, at the object in old and new states or at the *New Properties* all together or selectively. The “*Observe Centre*” is not limited to sight only but can get input from any of the five human senses, although current version of *CAGED* provides only visual link.

Once a command for particular change to object is issued, “*Command to Signal Converter*” coupled in series to the actuators make sure that it is performed. In some cases the latter two modules are actually part of the human body. However in general they are part of a *GUI* of a practical design system.

The sequence of events shown in the figure constitutes one complete design cycle and this cycle is continued until the design activity is finishedⁱ. Important points to realize are,

- One *NDC* loop occurs instantaneously (i.e. appears to *HD* as instantaneous, order of milliseconds)
- Since there appears to be no time lag between “*Object*” and “*Properties*” and since *HD* can observe them in any order, then *HD* can be fooled into thinking that the “*Object*” is the result of “*Properties*”. Forward and reverse analysis appear therefore to be equivalent.
- “*Good Hunches*” and the merit rules in the “*Merit Centre*” can be updated continuously at any time. This means that *HD* can change the direction of the design optimization and sense of the object at any time he desires.

The *HD* would be very lucky if all the processes in the *NDC* would occur inside the human brain (i.e. design activity would feel as natural as possible).because he is directly in touchⁱⁱ with the object , any change to the object is possible and instantaneous and new properties of the object are immediately available after any particular change to the object. However conceptual/preliminary engine models are complex. They contain large number of independent and dependant variables (i.e. higher than 3 dimensional which *HD* can visualize) and the analysis is too involved to occur within the human brain. Therefore the *Object*, *Analysis* and *Properties* have to reside outside of the human brain.

i. A colleague at TU Delft calls all the activity in figure 1 “analysis”.

ii. i.e. no need for an artificial “User interface”

This design cycle will feel natural to the human being if the following major conditions are satisfied

1. The design process is allowed to follow the natural path as described above. In summary, in the overall design system, it must be possible to,
 - Generate any object according to the continuously generated “*Good Hunches*” within the general domain of interest, and facilitate any change to the object that is likely to be required.
 - Employ a “User Interface Module” capable of understanding all the “Particular Change to *Object*” commands and one that can facilitate any necessary “Observation”.
 - Capability to analyse any “*New Object*” such that all *New Properties* of interest are generated.
2. The total chain of processes external to the human body must occur in “*Real Time*”ⁱ or in other words occur at similar rate to the design processes internal to the human brain and body. This similarity in time frames is called “*Real Time*” in this report. This implies that in a “*Real Time*” design Cycle, the time taken between issuing a command to “*Make a Particular Change*” and obtaining “*New Properties*” together with the time taken to “*Observe*” is so short that it can appear to be instantaneous to the *HD*.
3. Some aid is provided which when combined with the existing capabilities of the human, helps him visualize his multidimensional object in the format that he is very much comfortable with, the three dimensional geometrical domain plus time.

There remains one important question,

- why is it important for the design activity to feel natural? E.g. why should engine design activity feel natural given the fact that there are many engines flying and none of them were developed following the natural design path?

This question (or some variation of it), is the most frequently asked question at *CAGED* demonstrations to propulsion experts.

Natural Design path as described above offers several powerful advantages,

1. Forward and reverse analysis appear to the *HD* to be equivalent, because the *HD* can observe the *Object*, the *New Object* and the *New Properties* simultaneously

i. The term “*Real Time*” used in this report is not to be confused with that which is used in dynamic simulation of the engine.

or in any order that he pleases. For example in the case of aerodynamic design of aeroplanes, it is not important whether the analysis is set up such that it calculates the pressure distribution for a given airframe geometry or that it calculates airframe geometry for a given pressure distribution, because the human user can observe both airframe geometry and pressure distribution in *Real Time* without concerning himself with famous chicken and egg question. This equivalence offers great advantage because some problems naturally lend themselves to forward analysis and some to reverse analysis and in this system both can be used because they can appear to the *HD* to be equivalent.

2. Iteration is performed inside the brain and not outside. This means that the merit rules and the iteration variables are not fixed and that iteration decisions are made inside the brain and not outside. These are updatable whenever necessary as the design progresses. In this way it is possible for the *HD* to get and set the sense and direction of the optimization process and therefore remain aware of what is going on. Lack of this feel in the optimization process leads to large uncertainty in whether a global optimum has been reached.
3. The human being has very potent short term capabilities useful to the design activity which are mostly ignored if the design activity does not follow the natural path. Some of these are,
 - Short term vs. Long term memory. In long term memory, only the most significant pieces of information are stored. This section which contains the design experience of relevance to the current design activity, is mostly occupied with information other than that concerning the design activity. Long term memory in general is in general slower to access. However short term memory can store huge amounts of information with "*Real Time*" access necessary in natural design.
 - Pattern and Contrast Recognition. Using this capability it is possible to understand very large (if not unlimited) number of simultaneous influences and events.
4. In traditional approach to engine design, a designer has to possess a large amount of experience in order to reach a good design solution. While this approach has lead to innovations, it is limited because in the first place it takes a good designer many years to develop such experience and secondly the situations and conditions that made those experiences valid might have changed. Experience is in general not universally true and has a limited time span of applicability. In the natural design approach it is possible to gain valid experience relatively quickly, by making hundreds of effective¹ changes to the "*Object*" every minute and

i. "*effective changes*" are systematic and quickly considered changes i.e not random changes.

amassing the necessary experience. To illustrate the point consider the following propulsion experience whether it is correct or not,

- “for a two spool mixed flow turbofan, as the design flight Mach number increases, the overall efficiency of the engine increases, and the higher the design Mach number, the lower the optimumⁱ bypass ratio and the higher the optimum fan pressure ratio”

Now a novice propulsion student may typically spend days or longer to gain the above experience, but if the same student was using *CAGED* it is guaranteed that the above experience and much more is gained in one minuteⁱⁱ.

If one is convinced about the advantages of the above design approach, one may ask the following question,

- is it possible to build a design system now for gas turbine based engines that comes close to or follows the *NDC* described above?

In the absolute sense, true natural design is only possible if all the discussed processes occur within human body, i.e. it must be possible to visualize a highly dimensional object (e.g. a gas turbine) within the mindⁱⁱⁱ, and perform instantaneous analysis on this object. To the author's best knowledge, this capability is not yet common to human beings.

However, with today's work stations^{iv} with high speed CPU and high speed graphics, it is possible to closely approach a “*Natural Design System*” provided we restrict design to the preliminary and conceptual phases,

This restriction is necessary because in the conceptual and preliminary design, the idea is to include as many effects and influences on the “*Object*” as possible in order to fix it to within a reasonably narrow band of possibilities for further detailed design. In this regime, need for very high accuracy is outweighed by the need to include large number of design variables and merit rules. For weight and costs errors of 5 to 10 percent may be permissible. The quality of thermodynamic performance calculations however,

i. “optimum” here refers to maximum uninstalled overall efficiency.

ii. After initial learning of how to use the system.

iii. i.e. independent of devices external to the body.

iv. A 36 MHz. Silicon Graphics Indigo or comparable work station qualifies.

should be much better. For example, current version of *CAGED* delivers circa 99.9% accuracy for on-design thermodynamic performance for example.

It is today possible to make engine analysis routines which are instantaneous and contain weight, cost, dimensions and thermodynamics effects while fulfilling the preliminary design level accuracy requirements. It is also possible to aid the *HD* visualize a multidimensional "*Object*" and multidimensional "*Properties*", by allowing "*Real Time*" access to all the independent and dependent aspects of the "*Object*" and by combining "*Real Time*" animation and "*Real Time*" analysis routines. See section 2 for basic tricks on the subject.

CAGED system tries to mimic the "*Natural Design*" philosophy and the current version has been at least successful in demonstrating that certainly in the conceptual / preliminary design phases, this philosophy offers powerful advantages. This demonstration was performed on a 4 year old technology 12 MHz SGI Personal Iris Workstation which derives its high graphical speed from a separate "Graphics Engine" processor. The same workstation of current technology is now at least 4 times faster.

The Significance of "Natural Design" is not limited only to conceptual and the preliminary design phases, because as computing power increases, more detail of the object can be included in the "*Natural Design*" process.

If all the systematic problems of setting up a "*Natural Design System*" for designing propulsion systems is solved, then it is conceivable that more accurate and more calculation intensive analysis modules with more independent variables could be developed, as computing speeds and memory increase. It is possible to imagine that one day, if the workstation is many orders of magnitude faster than a current technology SGI Power Iris, it will be possible to attempt a full detailed design of complicated propulsion systems in say a few days.

After the lengthy discussion above the following statement also made at the beginning of the section should be meaningful,

- "*CAGED* is a design system which includes the *HD* as its most significant module"

FIG. 3 *CAGED* and *ADAS* interaction by *CAGED* supplying data to *ADAS* engine library i.e. one way flow of information.

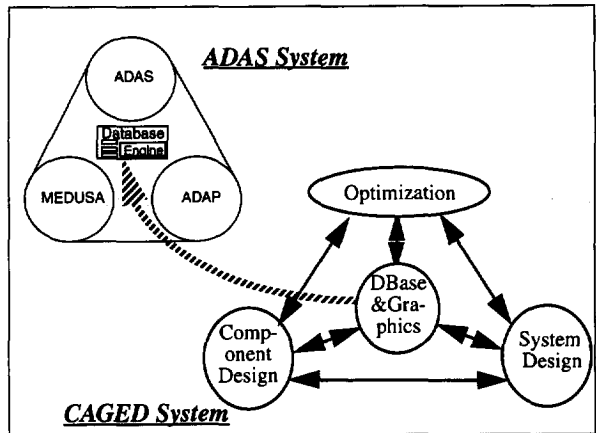


FIG. 4 *CAGED* and *ADAS* interaction by sharing common data locations. Two systems running in parallel. If *ADAS* issues a request for some data that does not exist a "data shortage" is recorded. *CAGED* will then notice this void and supply the necessary engine data

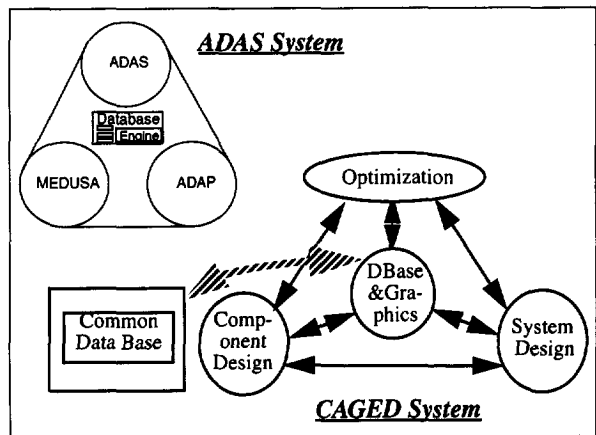
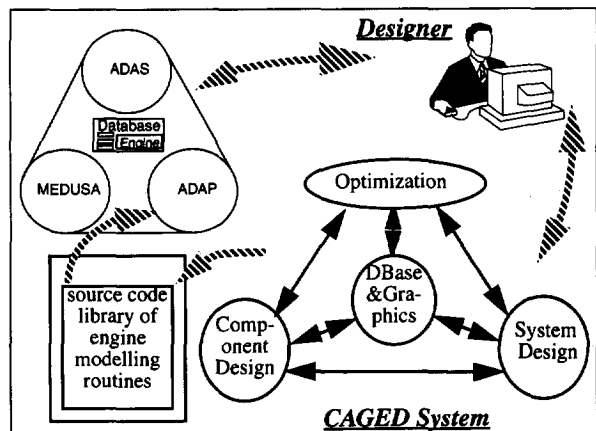


FIG. 5 *CAGED* and *ADAS* coupled by the *HD*. *CAGED* generates engine model source code for the engine configuration and variables of interest in the current study. *ADAS* will then use this source code instead of data.



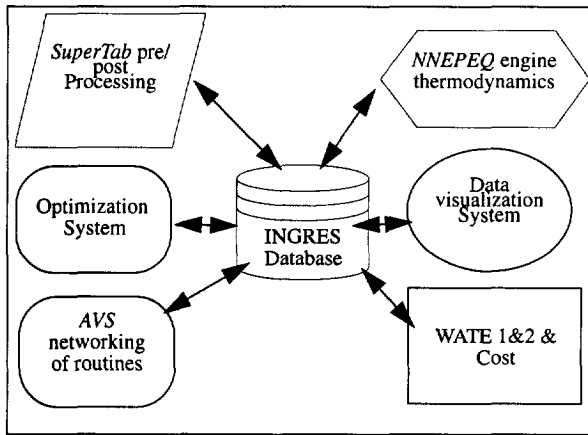


FIG. 6 An impractical system architecture for *CAGED* resulting from integration of several existing packages i.e. by not following an independent development strategy. Arrows indicate interface between major sub systems and the data base.

1.2 Motivation Behind Developing *CAGED*

1.2.1 Integrated Engine and Aircraft Approach

Traditionally the engine and air frame are designed and optimized separately and subsequently integrated to meet the requirements of the customer. Engine manufacturers often develop a new engine to suit a variety of aircraft which do not necessarily comply to the same mission requirements. It is then the job of the aircraft designer to make a choice between available engines for his aircraft. Although the two design processes are essentially independent, in order to achieve success in the design of the overall aircraft, both groups employ experts from the other side. For example an expert in the engine design group will typically try to understand the effect of engine design changes on the overall aircraft. Alternatively the aircraft designers typically may try to find which design modifications to the existing or conceptual engines will be of benefit to their aircraft.

This philosophy of separately developing the engine and aeroplane is also reflected in the majority of computer aided aircraft preliminary design synthesis programs existing today. The *Aircraft Design and Analysis System (ADAS)* ^[3] also developed at *Delft TU* possesses an engine library where performance data of a limited number of engines are stored. This limitation is due to a general lack of availability of engine data in a university environment. Simple scaling rules are then applied to engine data to obtain a new data set corresponding to the new thrust level necessary for the aircraft under current consideration. Apart from inaccuracyⁱ involved in this simple scaling, this method is very limited because the effect of major internal design choices of the engine

on the overall aircraft cannot be studied. In addition engine configurations for which no data is available or non existent cannot be included in the design process.

The idea of designing one specific engine for a specific aircraft (i.e. the integrated design approach) in order to achieve better overall performance is getting more and more attention as design and computational costs become a less significant proportion of total costs of aircraft plus engine. The downward trend in this proportion is mainly due to increased application of faster and cheaper computers in the design process coupled with more efficient algorithms and better understanding of physical phenomena. An example of this progress is the ambitious NASA NPSS (Numerical Propulsion System Simulation) ^[43] which aims at “all levels of the design process by coupling sophisticated simulation codes (e.g. CFD) and Hyper-computers to perform interactive, multi disciplinary simulations of complete propulsion systems such that there is only a single hardware-build-test prior to certification”.

The case for better overall performance of the integrated engine and aircraft becomes more urgent as the weight and range of aircraft increase. Table 1 shows that the influence of the power plant in the overall operating cost of aircraft is very significant particularly for the long range aircraft. It can therefore be argued that there can be gain by designing the engine and aircraft together if development costs of this approach become less significant by improvement in technology.

TABLE 1**Power plant related aircraft direct operating cost breakdown^[51]**

	Short Range 100 seats	Medium Range 200 seats	Long Range 450 seats
Overall = fuel + ownership + maintenance	35%	45%	55%

Towards this goal of integrated engine aircraft studies, the *Computer Aided General Engine Design (CAGED)* project was initiated in August 1989 so that it could in future be coupled with *ADAS (Aircraft Design and Analysis System)* also developed at the Aircraft Design and Flight Mechanics Group at the Aerospace Engineering Faculty of TU Delft ^[3].

-
- i. *Simple scaling here means simple thrust and weight scaling which is inaccurate. Accurate scaling is possible of course if detailed data of a wide range of engines is available and detailed scaling is applied at the component level.*

CAGED was to start life as an independent conceptual / preliminary design analysis system for gas turbine based engines. Once *CAGED* has reached an acceptable level of maturity there are plans to integrate it with *ADAS* or other aircraft synthesis programs. Three techniques have been identified to achieve Integration:

1. Let *CAGED* generate data for a variety of engine configurations and save it in the currently relatively empty *ADAS* library as shown in Figure 3 on page 10.
 - This option is the most simple but the least effective method because then *ADAS* will have no control over the engine parameters that generated the data sets in the first place. For example data sets were generated using certain component efficiencies and bypass and overall pressure ratios. If then *ADAS* in the course of some optimization studies decides to change say the component efficiencies or bypass ratio, then it has no way of regenerating the data. It is not possible to generate data that maps all current and future engine configurations and that includes all the variables of current and future interests.
2. Run *CAGED* in parallel with *ADAS* by allowing the two systems to have a common data base (see Figure 4 on page 10). In this way the actions of one system is allowed to influence the other through sharing of common data base or by transferring data between systems.
 - This is a fix to the first option. If the two systems share a common location in the data base, then if at any time a set of engine data is required by *ADAS* which is not available, *CAGED* can sense the data void and fill in the necessary data. This could prove very complicated to program, would require writing and continuously updating interfaces to a modern data base system, and would require that the two systems and the data base system run simultaneously on the same machine or via a network.
3. Include *CAGED* in its entirety as a sub system of *ADAS*.
 - Here it would be possible for *ADAS* to access all *CAGED* functions. For example at run time, in the optimization process of aircraft and engine, it will be possible to change configuration of engine and or select/reselect any engine variables as "interesting" parameters. This would be very difficult to program and *ADAS* is not able to provide this level of functionality. Furthermore the resulting total system would be very large, inefficient and slow.

4. Let *CAGED* generate a compact engine modelⁱ for the engine of current interest including all and only the independent and dependant variables of interest. This compact and efficient code can then be included as an engine subroutine and linked with *ADAS* or any large optimization packet as shown in Figure 5 on page 10.
 - With this method an engine configuration is defined and some independent engine parametersⁱⁱ are selected (e.g. bypass ratio, fan pressure ratio, turbine temperature etc.) and some dependent engine parameters (e.g. thrust, sfc, propulsive efficiency etc.) are selected or definedⁱⁱⁱ. For this particular case of current study an engine model file is generated by *CAGED* which includes thermodynamic and mechanical effects and gives the relation between independent and dependant variables. Here the engine configuration is fixed but all the engine variables of interest are available to the aircraft synthesis program by using the engine model as a subroutine. This makes any data base unnecessary since any new condition is recalculated.

The last option is the best trade-off between practical and functional consideration. In this way *CAGED* and *ADAS* are running completely independently i.e. at different times or on different machines. Since only the engine model is included within another program and not the whole engine design system, this method will be very efficient (i.e. in CPU time usage) and relatively simple to implement. Here the only limitation is that engine configuration and the set of interesting engine variables cannot be changed^{iv} from within the optimization loop.

-
- i. Here "Engine Model" is an ANSI C source code routine which is generated by the system for any given configuration and which includes all the independent and dependant and constraint variables of interest to *ADAS*.
 - ii. it is also possible to define an imaginary external parameter including a combination of engine design parameters.
 - iii. it is possible to select from a standard set of dependant variables (e.g. propulsive efficiency, specific thrust etc.) most commonly encountered or alternatively a new parameter can be defined including effect of any independent, dependant or other engine parameters.
 - iv. It is possible for *ADAS* to make a selection of available engine models. However these engine models are only generated by *CAGED* system. If it is necessary to change the engine configuration from within *ADAS* the only possibility is to select a new engine model from a library of available engine models. A new engine model that does not already exist in library cannot then be regenerated from within *ADAS*

Since the *HD* is responsible for all decision making, he is used here as the link between the aircraft and engine design activities i.e. to generate new engine models as and when this is necessary. The author is not aware of any aircraft design synthesis programs (with the *HD* as their key element) which are smart enough to attempt to change the engine configuration independent of the *HD*. This type of capability will only be necessary in systems using artificial intelligence concepts where it is conceptually possible to design with little or no human interaction.

The subject of integration of aircraft design with engine design will not be discussed any further in this report. Suffice it to say that *CAGED* was developed with this in mind and that the Aircraft *Design* Group has strong intentions to go in the integrated design direction after a certain level of maturity has been reached by *CAGED*.

1.2.2 Need for an Independent “Stand Alone” System

A common approach among engineers (not software specialists) developing applications is to integrate a set of existing large systems to satisfy the overall system requirements. This method has the advantage that the developer can have little or no knowledge of the internal features of subsystems that he is integrating. The only effortⁱ required here is to set up a central database and write an interface between each subsystem and this database. Integrating each major subsystem is only useful if and only if the majority of the functions offered by it are of significance to the overall system. Initial versions of *ADAS* for example used the *MEDUSA* 3D drawing and modelling package solely for defining the aircraft configuration/geometry and for generating 3D views. *INGRES* data base was also used at the centre of the system to couple the various parts. However both *MEDUSA* and *INGRES* offer a great number of functions of which *ADAS* only needs a few. Realizing this, current version of *ADAS* has completely done away with the data base and uses *AutoCad* instead of *Medusa*.

Figure 6 on page 11 shows an attempt to satisfy some of *CAGED* requirements by integrating several existing large systems. *NNEPEQ*^[14] would be used for thermodynamic calculation of engine networks and the *Boeing WATE1 & 2* code^[46] would be used for preliminary mechanical design of system. *SUPERTAB* part of the *IDEAS* package would then be used for pre-and-post processing, *AVS* (automatic visualization system) would be used for graphical definition of the engine network and as *GUI* to the system,.....etc.

It should be immediately obvious to anybody familiar with the above packages that the overall system as described in the figure would be impossibly large and inefficient. The

i. initial effort is small but in the long run becomes a huge effort

total system would not be possible to load locally on a down to earth work station with limited disk space, and would have to be implemented on a network.

Despite the fact that the overall system would offer great possibilities and flexibilities, these are not necessarily the ones that we are particularly seeking!

Another problem is that all these packages are being frequently updated. Most packages claim "upward compatibility" but this is not in all senses. By "upward compatibility" is usually meant that if a model or object was created by the previous version this can now be converted to the new format, however the new version works with this new format only. This implies that all the interfaces have to be rewritten each time one sub system is upgraded. Some times the upgrade may be so severe that changing the interface is not enough and the set up of the data base has to be redeveloped.

Economic and availability considerations also play an important role in this respect. All software costs money and royalty fees may be involved if other software is being used integrally. Those that are cheap may not be available to everybody.

An independent "stand alone software" which is purpose developed for a set of particular tasks is very compact, efficient and cheap. Here, only one interface to the operating system of the computer is required making the total system immune to system managers, network failures, continuously writing and updating interfaces...etc, in a less than perfect computing environmentⁱ.

1.2.3 Need for General Approach to Engine Configuration and Fast Engine Models

New engine concepts or modifications to existing engines in the conceptual and preliminary design fields are very frequent in the quest for the most efficient or suitable configuration. Reference 66 gives a review of engine design trends for the next 50 years. Apart from the expected evolutionary improvements in current concepts due to better materials and better physical understanding of flow and heat transfer phenomena, a lot of work is being done to study the effect of various cycle configurations. The configuration possibilities are almost endless from propfan to turbo-ram-rocket to liquid air cycles to cycles using stationary and non-stationary detonation waves in the combustion chamber or as after burner etcetera.

i. "perfect computing environment" is one that employs professional system managers and some thought has been given to selection of hardware and software and the overall set up of the computing environment.

Early engine analysis programs were developed for fixed cycles, but soon the general approach took over as the number of interesting cycles grew and as computing power/price ratio improved. See section 3 .

The generalized approach is much more difficult to implement than the specific approach to engine configuration, however this initial effort is preferred to developing and re-validating new engine codes every time the concept is changed or even slightly modified. This approach has been successfully implemented for several years now, e.g. [14] claims to handle “most, if not all” configurations.

The generalized approach however, usually results in large engine routines that are slow to execute.

Aircraft preliminary designers, or people who need to include an engine routine within some larger optimization program, usually prefer “*Engine Decks*”ⁱ because these tend to be much more compact and faster to execute.

One of the challenges in the CAGED Project has been to combine “*High speed Engine Modelling*” and “*Generalized Engine Configurations*” effectively. The current version of CAGED has so far been successful in achieving and demonstrating this combination in the thermodynamic sense.

1.2.4 Need for Real Time Optimization Methods

This is one of the strongest needs present in any design group in general and in aircraft preliminary design in particular. Aircraft designers must typically handle a crowded pool of design variables and as discussed before the human brain has difficulty visualizing any “*Object*” that requires more dimensions than the physical 3D Geometrical domain. In practice this means only three independent variablesⁱⁱ.

Due to this limitation, the object with large number of variables are usually fed to some optimization routine and the involvement of the *HD* is limited to the initial set up of the problem. Not involving the human within the optimization process causes lack of awareness and may lead to non-optimum solutions.

-
- i. “*Engine Deck*” or “*Cycle Deck*” are engine routines specifically written for the specific engine under consideration, usually provided by engine manufacturer to aircraft manufacturer.
 - ii. This minimum can be increased by perhaps two by clever combination of colour and lighting to the 3D physical geometry domain.

Mimicking the “*Natural Design*” as described above, will keep the *HD* fully aware of his “*Object*” and its “*Behaviour*”. This leads to quick build up of valid and current experience about the “*Object*” and more certainty in the solutions.

1.3 SCOPE of the *CAGED* engine models

CAGED is a conceptual preliminary design and analysis tool for gas turbine based aircraft engines. The concept can be classified as an *Integrated Thermo-Mechanical Design* system complying to the *NDC* philosophy when fully implemented. Engine configurations are treated as a network of standard components. As will be fully described in this report, *CAGED* generates an engine model for any engine configuration in which the flow of information is towards downstream. This is a network for which the “*Serial Network Logic*” is applicable. See section 4.2.1 .

The *CAGED* engine models in their full glory include the following effects,

1. generality in configuration of the engine
2. steady state on-design thermodynamic performance
3. steady state off-design thermodynamic performance
4. quasi-steady transient thermodynamic performance
5. determination of critical conditions (e.g. maximum shaft speeds, maximum temperature etc.) for each component by successive calls to the off-design model for each altitude/Mach pair that define the mission/ flight envelope.
6. determination of component dimensions, weight and cost from the critical conditions of the previous step
7. automatic or user interactive assembly of the above dimensioned components into an annulus drawing of the engine (i.e. a cross-sectional drawing showing how the gas path fits with the geometry of the components)

All the methods above are in principle component based. The integration of *thermodynamic* and *mechanical* design is achieved by going *from* component thermodynamics *to* component geometry so that any change in thermodynamics of the engine is instantaneously reflected into geometry of the engine. This approach is discussed in detail in chapter 5 with a systems drawing of all the necessary procedures and modules. The chapter also describes the reverse approach, i.e. going from geometry to thermodynamics, but as will be explained in that chapter, it is too difficult to set up and use. It has been explained in the preceding chapters that there is an apparent equivalence between *forward* and *reverse* analysis in the *NDC*, due to the high execution speed. Therefore the *forward* alternative is chosen which is much easier to set up and use.

It was unfortunately not possible to implement the *CAGED* system in full, due to lack of time. However all the theoretical basis and know-how required to achieve a full implementation has become available in the current research program. All the necessary *thermodynamics* (steps 1 to 5 above) was developed by the author. The rest (steps 6 and 7 above) would have to be borrowed from the references published on the *Boeing WATE*^[46] [47] code which was developed under contract from *NASA LeRC*.

Once an engine configuration and variables is defined and an engine model is generated by the *CAGED* system, then within this specific engine model the following two aspects are *fixed*,

- Engine Configuration
- Mechanical Design Rules

However since the process of regeneration of an engine model is very efficient and relatively quick, *CAGED* can claim to have a general approach.

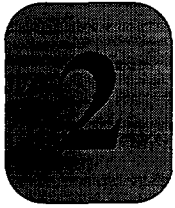
The generated engine models are high speed and in the form of uncompiled *ANSI C Source Code*. These can be included within other programs as subroutines or can become part of a “*Natural Design Cycle*” when included within “*DynCarp*” and “*DynHist*” programs. See section 2.2 for detailed description of *DynCarp* and *DynHist*.

1.4 System Requirements Summary

The following section summarizes the challenging requirements that were set for *CAGED*.

1. Independent “Stand Alone” System
2. Full *GUI* to all System Functions
3. Generation of Compact High Speed Engine Models suitable for inclusion within the *NDC* or within large Synthesis or Optimization Programs
4. Representation of Universal Component as a Mechanical and Thermodynamic Unit
5. Full set of Standard Components
6. A Network Logic that can handle general thermodynamic networks.
7. *GUI* to access all system function and model data
8. Include On & Off Design Thermodynamic Analysis, Sizing, Weight and Cost in the generated Engine Models
9. possibility to use any fuel and oxidizer in the system
10. All system functions accessible without need for Programming

11. New methods in optimization and model exploration to aid the *HD* make sense of crowded pool of design variables
12. Capability for Future Integration with *ADAS*
13. *CAGED* must be a "Total System" that performs all necessary functions but must be modular in construction so that each major sub-module can be used or updated separately or used as part of other systems
14. *CAGED* must mimic the "*Natural Design*" (See Fig. 1) as closely as possible.



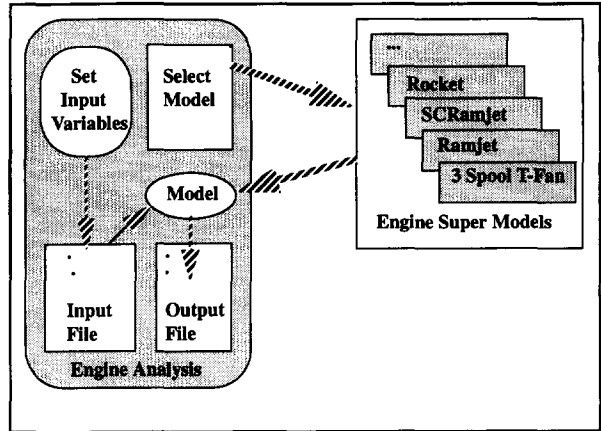
Basic Approaches and Tools for obtaining “Real Time” and “Natural Design Cycle”

As explained previously one necessary condition for a “Natural Design Cycle” is that the total chain of processes outside of the human body appears to occur instantaneously to the human brain. The speed and functionality requirements are very challenging to meet together. Hence the inclusion of this chapter.

The traditional engineering approach to modelling, concentrates on achieving high accuracy by the application of brute force. It is usually very difficult for engineers to appreciate methods that provide say 95% accuracy in a fraction of a second as compared to methods that provide 99.9% accuracy in say sixty minutes. Often the former involves more artful application of theory and empirical results. Accuracy is a vague term and is a difficult basis for comparison between various methods. For example a thermodynamic package claims thermodynamic accuracy of 99.99% for any general engine network. A conceptual engine design package may claim thermodynamic, mechanical, and installation accuracy of 90% altogether. The former system is not necessarily more accurate than the latter, because different level of detail and different amount of overall effects have been considered, i.e. we are not comparing similar things.

Development of *CAGED* involved including as many influences as possible in the final generated engine models while remaining within a level of accuracy that is acceptable to a Conceptual / Preliminary Design Package. In many cases the objectives were achieved without any compromise on accuracy only by improving the concepts and the set up of the problem. In other cases new analytical approach was needed that meant some compromise on accuracy but which gave boost for functionality.

FIG. 7 Traditional Approach to Engine Modelling, by developing several super-models where each super-model represents a class of engines by parametric manipulation.



One general way to gain vital speed is,

- Try maximum amount of processing possible before creating the model

Here we try to maximize all the processing before the engine model is created if this results in a saving in processing within the engine model. See section 2.1.5 for example of this.

Now *CAGED* will be a complex system and traditionally complex systems are bulky and inefficient to run. Since we are not allowed to change the system requirements we can rearrange this complexity by,

- Try to hide the system complexities somewhere other than the place where they could cause reductions in speed i.e. not in final engine model.

The latter requires some conceptual changes to traditional approach to engine modelling and analysis i.e. it is possible to remove “Network Logic”ⁱ from the final engine model that is created and discard any variables that are not of interest to the current study. See section 4.2.1 .

In some problems it is possible to avoid bottle necks by,

- developing new analytical or semi-analytical short-cuts

i. “Network Logic” is method by which a general thermodynamic network of components can be calculated. See section 4.2.1

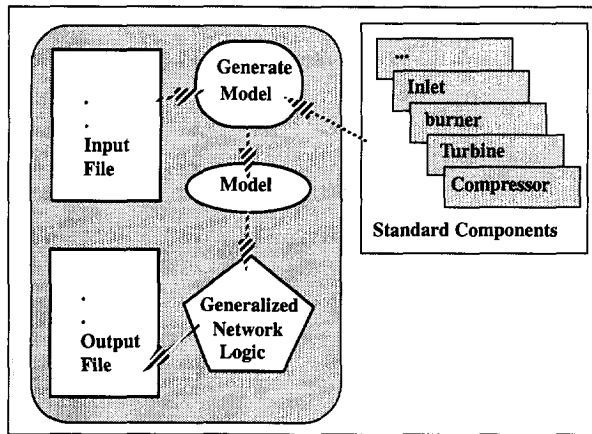


FIG. 8 Generalized Approach to Engine Modelling & Analysis. A new engine configuration is defined by connecting a set of standard components into a network. The Network Logic module can handle any such network. The model includes all the variables that may be required for all types of engines.

The level of accuracy required from a conceptual/preliminary design system, allows in some cases direct solutions to problems which are iteration intensive. In off design thermodynamic calculations for example, analytical effort lead to a direct steady state off-design method for a generic spool with 96% to 99% percent accuracies.

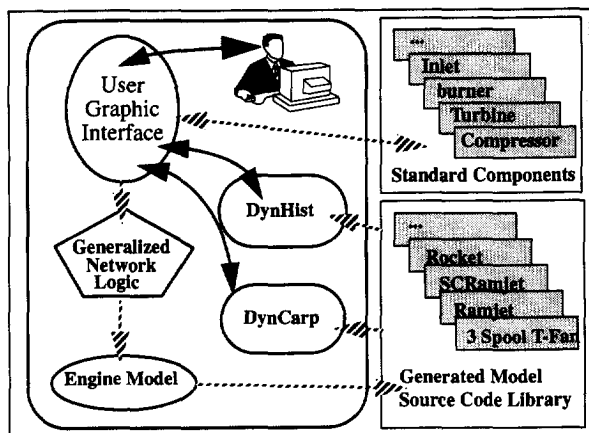
Other more obvious efforts,

- Choice of Correct Programming Language
- Follow efficient coding practices

ANSI C was chosen because of the following

1. *CAGED* is extremely function call intensive and *ANSI C* has minimum overhead associated with function calls on a *UNIX* machine
2. *CAGED* tasks are not exactly predefined for each and every run. Array sizes, and functions to be called, are not always fixed in advance requiring allocation at run time. and use of pointers. e.g. the maximum number of components in an engine configuration is not fixed and space is allocated after every component addition. The use of pointers and dynamic allocation is particularly convenient to handle in the *C* language
3. *CAGED* tries to use capabilities of the workstation to the limit and *ANSI C* is very close to the *UNIX* operating system
4. Object orientation is a facility that is well suited to modelling and creating new "Objects" as and when necessary. Currently rudimentary Object Orientation is manually programmed into *CAGED* for modelling and recursive sub-modelling or super-modelling. However in future *C++* may be used as this makes classification and scope of the large number of variables in the source code more convenient to handle as well as making it easier to integrate modules developed elsewhere into the system.

FIG. 9 CAGED approach to engine modelling. Source Code is generated for engine models depending on the Human Designers choice of variables and the engine configuration that he defines. "Network Logic" is used to generate the models but is not included in them. Solid arrows indicate "Real Time" two way link and hashed arrows indicate normal one way link.



Although *Fortran 77* is language of choice for many engineers who write calculation programs, *ANSI C* offers the best flexibility to programmers developing large multifunctional systems.

Efficient coding practices can also lead to time savings but in much less significant amounts compared to conceptual and analytical efforts.

2.1 Conceptual Changes

2.1.1 Engine Modelling

Traditionally engine modelling and analysis tended to be specific to particular engine classes as shown in *Fig. 7*. e.g. *GASTURB* [36]. This approach has the advantage that each model is compact, efficient and simple to develop. Since analysis methods and data are much more easily available for fixed engine types, it is easier to develop a relatively accurate model in a short time. Each engine model represents a variety of engines by parametric manipulation and switches. e.g. setting bypass ratio of a turbofan to zero and fan pressure ratio to unity reduces a 2 spool turbofan model to single spool turbojet. This approach offers very little flexibility since engine parameters can be set and not selected. Also modifications to model beyond parameter manipulation is not possible and any new engine class has to be reprogrammed and revalidated.

The need for general approach to engine configurations has resulted in systems that can be represented by *Fig. 8*. Here a full library of standard components exist which can be used to set up a network that can represent any conceivable engine configuration. E.g. *NNEPEQ* [17] or *MOPS* [35]. All independent and dependant variables are available in

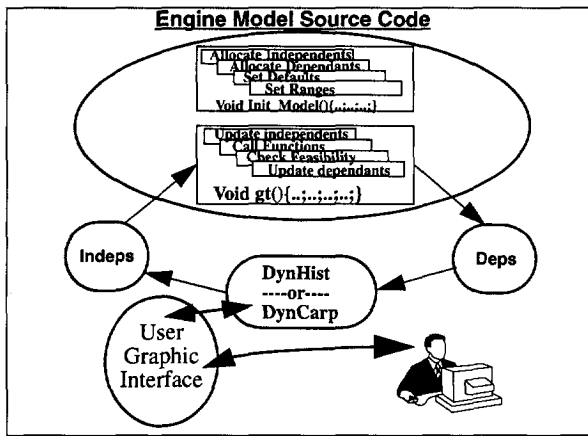


FIG. 10 Engine Model in *CAGED* is an uncompiled *ANSI C Source Code* which is generated for any user-defined configuration and independent and dependant variables of relevance to the current study/design. The file consists of two functions, *Init_Model()* and *gt()*. The former contains mostly allocation and initialization information of variables which need to be done only once. The latter function is a high speed transfer function between dependent and independent variables. Arrows indicate "Real Time" Link

the engine models that are generated i.e. variables cannot be selected but can only be assigned a value. The Input File contains information on how components are connected and values of all the variables. The heart of this approach is the Generalized Network Logic. This module is set up to solve any model configuration that is fed to it. Every time independent variables take on different values, the model in its new state is fed to this module. This approach is therefore much more difficult to develop but in the long run saves a lot of time and effort as new engine concepts arise and as current concepts need to be modified.

As with all cases "Generalization" is very difficult and not totally absolute. It will always be possible to think of some type of engine concept that cannot be solved with current level of generalization. Systems that generalize tend to require improvement in the level of generalization with time.

The latter type of system is powerful but tends to be large and sluggish and not suitable to "Natural Design". The slowness comes from the fact that the system has to "figure out" how to solve the engine network every time the state of the engine model is changed, i.e. every time an engine parameter is changed. But "figure out" is necessary only if the actual engine configuration is changed. Change of engine configuration is a task that occurs far less frequently than parameter variation. The generalization effort is therefore actually wasted every time an engine parameter is changed.

To obtain speed and generality together, *CAGED* takes the generalization out of the engine model that is created. Fig. 9 shows the set up for such a concept. The human designer makes the following steps in sequence,

1. With the aid of the "Graphic User Interface (GUI)" he,
 - defines a network of standard components

- defines dependent and independent variables of interest to the current study. These can be any mathematical combination of all the parameters internal to the engine and arbitrary external variables.
 - defines type of effects included in the model file, e.g. On- or Off-Design steady state thermodynamics only
2. This information is then fed to the “*Generalized Network Logic*” which generates an engine model. This engine model is in the form of uncompiled *ANSI C Source Code* as shown in *Fig. 10* named “*Gt.c*”. In order to gain speed, the initialization, allocation and other information which need to be done only once are written in a separate “*Init-Model(){...;...}*” function. All processes and information that needs to be updated is included in the *gt(){...;...}* function which has access to library of thermodynamic and mechanical routines and will call the necessary functions in sequence, as and when necessary. There is therefore no time wasted to “figure out” where to go next or which function to call. See section 4.2.1 for the limitation concerning the generalization method. It is important to note that at this point configuration, selection of variables and type of calculations is fixed and modification to any of these requires the human user to restart at step 1 above. The “*Gt.c*” model file generated is more compact and efficient than those in *Fig. 7* because only the necessary variables and the necessary tasks are included which are sufficient for current study and no more.
 3. Until now file “*Gt.c*” is not compiled and can now be compiled and then linked to “*DynHist*” or “*DynCarp*” or any other optimization program which can vary the state of the model by changing the values of independent variables. The function *gt(){...;...}* then provides the new values of the dependants instantaneously and will indicate whether some constraints were satisfied or not.
 4. The human designer then continuously makes quickly considered changes to state of the model by turning a dial for example. This change is reflected to the engine model and the calculated results are instantly displayed by “*DynHist*” or “*DynCarp*”.

The *ANSI C* code generated for the engine model is then *fixed* for the engine configuration and variables of interest for the human designer. Now what happens if the *HD* requires variable geometry in the engine model? If the type of variation can be anticipated in advance (e.g. the *HD* knows that engine of current study can vary from a two spool unmixed turbofan to a single spool turbojet, nothing more) then the *HD* can define this visually prior to source code generation.

For the example case, the *HD* defines a 2-spool unmixed turbofan. Then Externally Applied Relations *EAR*’s can be used for example, to make the bypass ratio as a function of flight mach number and altitude, in order to schedule the variation of bypass ratio of the engine. *EAR* is an artificial mathematical relation (as allowed by the C

Compiler) which can relate any variable of any component to any variable of other components, as long as the information flow is forward (i.e. towards downstream or towards back of the engine).

The above example is very simple. If geometry variation is very complex then the user defines all the “pure” cycles separately on the screen where they all get fed from a single mass source via splitters. The amount of mass flowing through each engine (i.e. the bypass ratios of splitters) will then determine to which capacity each engine is working. These bypass ratios can be stated in terms of mach number and altitude or another external variable by using *EARs*.

So in *CAGED* all possible variations required in the engine model should be provided by *HD* in advance. This is *not* a disadvantage in design systems where the *HD* plays a central decision making role, i.e. since process of regenerating engine models is quick and simple, a new model is effortlessly generated any time the *HD*’s prior anticipation is not sufficient.

However if the generated engine models are to be used within other optimization programs, then one must make sure that the model contains all the variation that is likely to be required.

2.1.2 High Speed Function Generation from Data

Function generation from data is very intensively required by design or analysis systems in general. One typical example is in the traditional iterative Off-Design Engine Performance approach. Here a “performance map” (i.e. a multi dimensional data file) is associated with each component. For every iteration these maps are interrogated to see the new state of each component and therefore the engine.

The topic of high speed function generation from data is currently intensively researched and has lead to the development of new hardware and software. Hardware development includes

- Low cost high speed memory blocks. See reference 59 for a propulsion application
- Programming function generation operations onto special chips^[30]

Linear manipulation of multi-dimensional data, i.e. “*Multivariate Newton Raphson*” is the simplest and most flexible generation technique and can be very accurate if the data grid is fine. However, the speed of this method decreases as the number of dimensions and the amount of data increases. For example consider a data file with only 4

independent variables (i.e. 4 dimensional data set) where each variable takes on 10 values. The resulting data set will have 10^4 data lines. Simple linear manipulation of this data set will be too slow for application to *Natural Design*.

The sluggishness is caused by two simple procedures

1. Trying to locate the multidimensional grid that surrounds the required data point.
2. calculating a multidimensional gradient matrix to use for interpolation.

The former can be totally removed and the latter can be heavily accelerated if we adopt simple numerical tricks and modify the original *Multivariate Newton Raphson*. The trick is basically to transform the problem so that the table indices are considered as the independent variables of the table and the old independent variables can be considered as dependant variables. See section 6.6 for detailed derivation.

2.1.3 High Speed Optimization and Solution to n simultaneous nonlinear equations

Engine design and analysis systems rely heavily on Multivariate Optimization and solution to nonlinear sets of equations, especially in the following areas,

1. *Chemical Equilibrium Calculations* to calculate properties of the working fluid.
2. *General Thermodynamic Off-Design* approach to determine equilibrium running lines.
3. *Optimization of the Engine Thermodynamic Cycle*.

Since the sets of equations are generally non-linear, and highly dimensional, the solution schemes tend to be iterative. Typical of such approach is the *Multivariate Newton Raphson* (See section 6.6.2). This method is very general and can converge quickly if the first guess is good and the equations are not too non-linear in the neighbourhood of this good guess.

However, iterative optimization schemes are inherently sluggish and it is not possible to fix the calculation effort in advance for a given required quality in the result. In other words it is not possible to know how many iteration will be required for finding the true optimum in advance. The calculation effort is problem dependent and only properly known once the solution has been found.

In high speed applications it is highly desirable to fix the calculation effort¹ in advance for given required accuracies and required probability that the true optimum has been

found. For example it would be desirable in a high speed application (e.g. *CAGED*) to state the cycle optimization problem as follows,

- find the optimum (i.e. maximum overall efficiency) combination of design point fan pressure ratio, bypass ratio, overall pressure ratio and turbine entry temperature to within an error of 1% and certaintyⁱ of 95% that this is indeed the true optimum, further check that this can be done in less than 10 milliseconds on a benchmark workstation *prior* to commencing the search

This problem can be solved by systematic application of brute force (i.e. the *Non-Random Adaptive Grid* method) for the special case where the engine performance can be evaluated at high speed. Fast function evaluation is an inherent quality of high speed applications such as *CAGED*.

2.1.4 High Speed manipulation of accurate data from other sources than *CAGED*

There exist many engine analysis programs which are too slow to execute but which give valid and accurate results. This sluggishness implies that these routines, as they are, cannot be integrated within a *Natural Design Cycle*. However all is not lost because the data generated from these routines can be manipulated by high speed function generation methods discussed above.

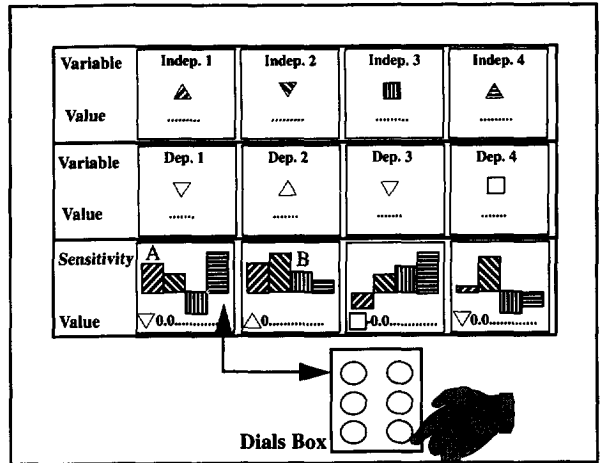
2.1.5 Fluid Properties from Chemical Equilibrium Calculations

In order to achieve higher thermodynamic accuracy, it is necessary to include the effect of dissociation of the working fluid, in engine cycles that use high temperatures. Reference 17 states that for a 1-spool turbojet, dissociation causes an increase of 11.1% in fuel flow, 6.4% increase in specific thrust and 4.4% increase in TSFC, with turbines running at stoichiometric temperature for JP4 fuel and air mixture.

NNEPEQ integrates *CET 86* (a modified version of *NNEP*^[25]). This provides a strong capability since the latter is written in general terms and any combination of fuel and

-
- i. *Calculation effort* is the total time taken on a particular hardware to converge to solution
 - i. *Accuracy* of the solution found and the *Certainty* in the solution found are essentially different concepts. See section 6.1.1 for mathematical descriptions.

FIG. 11 *DynHist* tool allows the human designer to move through the design space. Turning a dial changes the value of an independent variable. The resulting change in dependant variables and the sensitivity histograms are then instantaneously displayed.



oxidizer can be considered. The working fluid can be frozen or allowed to react throughout the engine.

The disadvantage of this approach is that the code becomes extremely sluggish because each time the routine is called, at every station, a new set of nonlinear chemical equilibrium equations have to be iteratively solved. Reference 17 states that including chemical equilibrium in the thermodynamics increased calculation time by factor of nearly 24 times.

While this penalty factor can be improved by "techniques to reduce number of iterations for convergence" real improvement comes from conceptual change. This change comes from the realization that during most computer runs, the type of working fluid used by the engine is fixed. For example, while studying the *air-turbo-rocket*, we may fix the working fluid to be a reacting mixture of Hydrogen and air for all runs. For more complicated study, we might fix the working fluid to be a reacting mixture of three given fuels and three given oxidizers. Typically, in engine analysis efforts, the types of fuel and oxidizer present in the engine are not independent variables and are actually fixed. One can then use this fact to devise a high speed method for generation of fluid properties.

If the basic constituents of the working fluid are fixed, it is possible to make only one run of *CET86* for all temperatures, pressures and mixture ratios of interest, *in advance*, generating a multidimensional data file, where the number of data lines are determined by the required accuracy and the range of temperature and pressures of interest. If we have one fuel and one oxidizer, this results in a 3 dimensional data table with pressure, temperature and mixture ratio as independent variables¹.

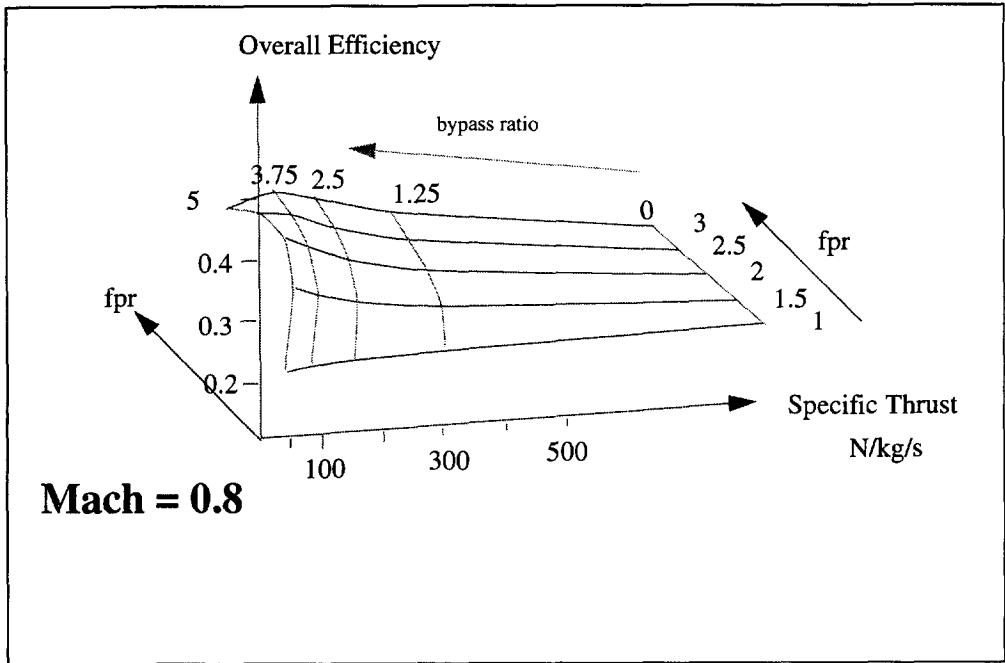


FIG. 12 Overall Efficiency vs. Specific Thrust as function of fan pressure ratio and bypass ratio for an unmixed two spool turbofan, as displayed by *DynCarp*. the carpet can be rotated about the three axes and zoomed in or out with a perspective option. This helps the user get the best view of the carpet.

2.1.6 Non-iterative Off- Design Methods

The traditional approach of solving the Steady State Off-Design thermodynamics equations, is to formulate a set of n simultaneous equations in n unknown engine variables, coming from continuity of mass, momentum and energy in the engine. The solution to these is then found by iteration of engine variables, e.g. by using *Multivariate Newton Raphson* (See section 6.6.2). It is possible to perform this procedure digitally at high speed by using specially developed computers^[59] or current fast mainframe computers and/or analogue engine models^[60]. On down-to-earth workstations, however, this iter-

- i. two fuels and two oxidizers would result in a 5-dimensional data table, where 3 mixture ratios, pressure and temperature will be the independent variables.

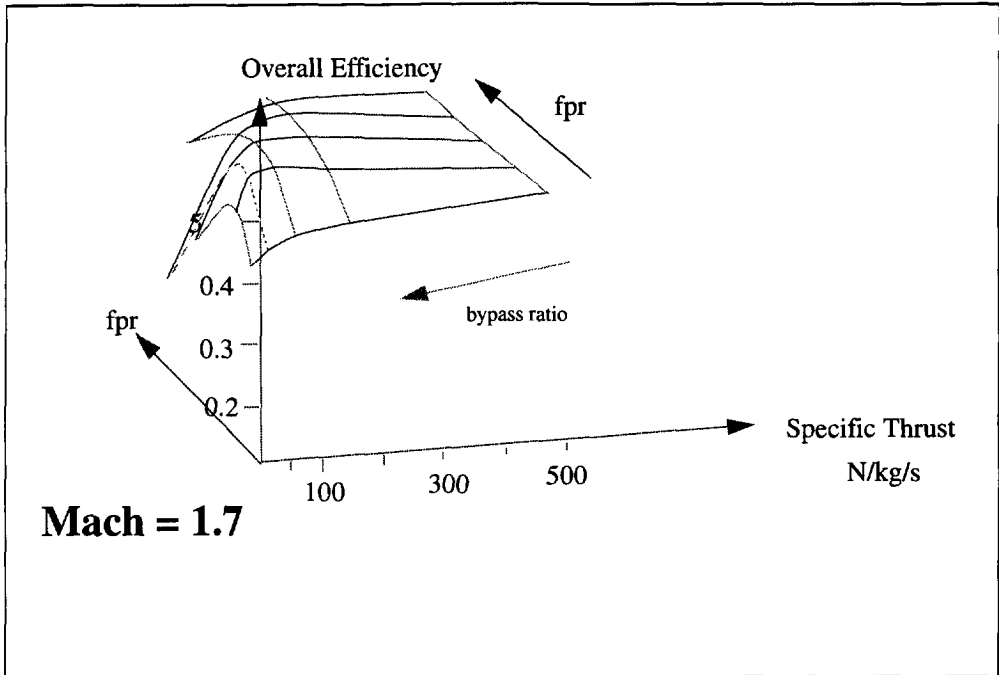


FIG. 13 Increasing mach number moves the carpet in the design space and changes it's shape. Note that the points on the carpet which are infeasible (infeasibility functions are defined in the engine models by setting a flag) are not included and are chopped out.

ative procedure is too sluggish for the *Natural Design Cycle*. An important point to note is that *Off-Design* is not the only issue here, and *CAGED* aims to generate engine models that include *On- and Off-Design* and *Mechanical Design* altogether, requiring therefore that each be performed at very high speed.

The alternative is (semi-)analytical methods which are non-iterative but which provide sufficient accuracy (96% to 99%) consistent with the Conceptual/Preliminary design phase. These fall into three categories,

1. **Global Analysis.** These are (semi-)analytical relations derived from the thrust equation and dimensional analysis. The engine is treated as black box and the internal configuration and details are not considered^[63]
2. **Choked Turbines or Nozzles.** Direct solution to the off-design equations by assumption that the internal spools are operating choked. The outermost spool is

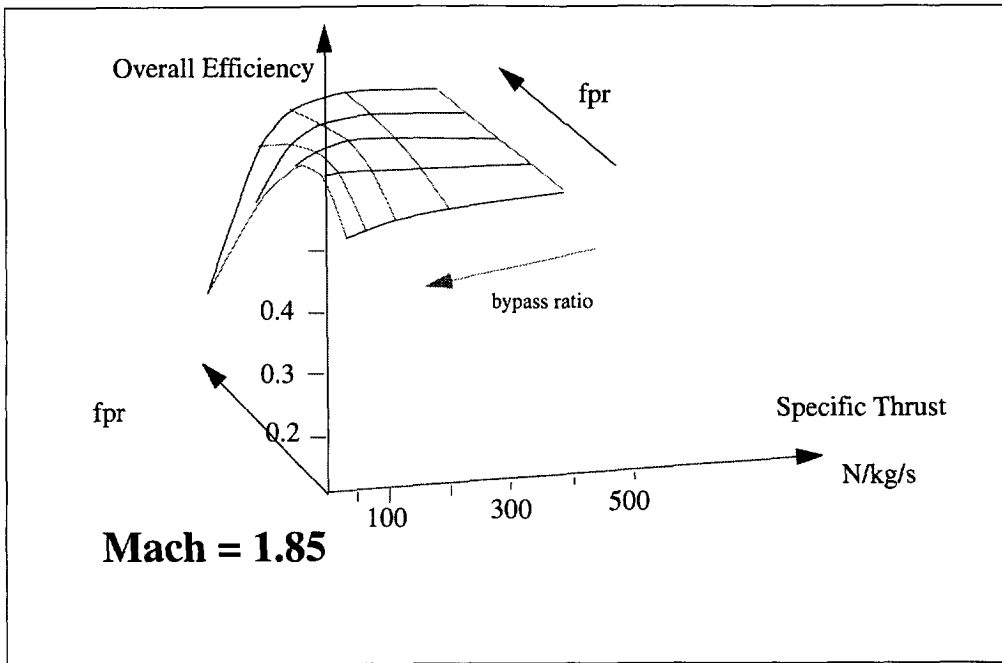


FIG. 14 Range of independent values can be adjusted in *DynCarp* also by turning dials. This can be used to concentrate or stretch the carpet in the independent direction. Here the highest bypass value was reduced, so that none of the surface was chopped, as it was in the preceding figure.

then either assumed choked (giving a direct solution) or coupled to exhaust nozzle characteristics (giving iterative solution with one iteration variable)^[68]

3. **Differential Analysis.** Differentiation of spool off-design equations^[65] and analytical manipulation^[7] gives sensitivity of spool variables with respect to the corrected mass flow parameter at inlet to the spool

The first approach is actually not useful to engine design because internal configuration of engine is totally ignored. It's use is limited to aircraft studies where very little detail of engine is considered. The second approach is surprisingly accurate, but is unfortunately only closed when turbines and/or nozzles are choked. Also method does not easily lend itself to *Sequential Network Logic*.

In this report a *Non-Iterative Closed Form* solution is derived (falls into the third category above) and validated for a *Generic Spool* (with bypass) which does not rely on the usual choking assumptions. See section 4.4 .

Further a very simple and *Non-Iterative* procedure is developed, which needs only the *Equilibrium Running Line* and the *Current Operating Point* to calculate the transient performance for the generic spool. Hence no need for iteration and or any component maps. See section 4.5 .

2.2 Real Time Movement Through Design Space

Usual approach to design optimization is that the human designer first defines the model, i.e.

1. Independent (or Design) Variables are defined, and so are their range of values
2. Dependant Variables and Optimization Functions are defined in terms of the above and each other
3. Equality and Inequality Constraints are defined, in terms of Independent and/or dependent variables

Then the designer feeds this model to a *Numerical Optimizer*, which after number crunching comes up with an *Optimum* point. The disadvantages and limitations of this approach are as follows,

1. The *calculation effort* (i.e. computation time) of almost all numerical optimization methods become *enormous* and impractical as the dimensionality (i.e. number of design variables) increases. Preliminary design involves usually large number of variables, 25 or 50 or more
2. There is usually no indication of how good this so called optimum is, i.e. most methods (the *Adaptive Grid* section 6.1 on page 119 excepted) don't indicate whether a *local* or a *global* optimum is found
3. Most numerical optimization routines, put a restriction on non-linearity, continuity, dimensionality of optimization function or the constraint functions
4. The numerical optimization process does not directly increase the understanding of the human designer about the problem, unless he is very experienced / knowledgeable about the *object* and can assign physical significance to a numerical result
5. After an *Optimum* is found, The designer may realize that the original range of variables was not sufficient requiring adjustment and rerun. This is a problem especially if the first run took hours or days of computation

In the specific domain of *High Speed Models* (or high speed optimization functions), it is possible to *Move Through A Highly Dimensional Design Space* in *Real Time*, as an alternative method for model optimization. Here there is *no real limit* on the *dimensionality* of the model and also no restriction on the type of optimization or constraint functions (except the speed). In addition, since the human designer is active in the optimization loop, *Understanding* is accumulated about the multi-dimensional model behaviour which aids in finding a *True Optimum*. The following sections describe the *DynCarp* and *DynHist* tools which are general (i.e. not only applicable to engine design) and which take advantage of high speed engine models generated by *CAGED*.

A general problem with highly dimensional domain, is the *Visualization*. The human mind needs to be aided for visualization of higher than 3 dimensional models. *DynCarp* and *DynHist* aid the designer to some extent, but not if the carpet plots or histograms vary in an irregular or unpatterned way (e.g. if the curves of the carpet plots are all crossing and jagged) through the design space. For this reason the *Non-Random Adaptive Grid Numerical Optimization Method* (section 6.1 on page 119) was developed which is especially suited to multidimensional high speed model where the dimensions are not too high, i.e. less than 25 on a benchmark workstation.

2.2.1 Dynamic Histograms “*DynHist*”

Fig. 11 visualizes the *DynHist* program. The user sees three windows on the monitor. Top, Middle and Bottom windows display the independent variables, the dependant variables and the sensitivity histograms respectively. Each dial is coupled to the value of an independent value. Clockwise turn increases value, and anticlockwise turn reduces value. Increase, decrease and no change in values are displayed as upward pointing triangle, downward pointing triangle and square respectively.

The sensitivity bars display percent change in each dependant variable with respect to one percent change in each independent variable. These bars are colour coded (here shown as different shading) and positioned to show which independent variable each bar represents. For example bar *A* represent sensitivity of dependant variable 1 to independent variable 1, bar *B* represent sensitivity of dependant variable 2 to independent variable 3. The text values displayed in the sensitivity window displays the sensitivities with respect to the independent variable currently being changed.

DynHist follows these simple procedures,

1. the user changes the state of model by turning a dial

2. the Independent variable connected to the dial changes value instantly. Model function $gt()$ is called to get the new values of dependent variables
3. a small perturbation is applied in each independent direction separately, resulting in calculation of sensitivity values
4. The new situation is displayed in all three windows
5. Going from step 1 to 4 is *instantaneous*. Steps 1 to 4 can be continuously repeated in any independent direction

It is at first difficult to get used to this application especially if the number of independent and dependant variables is large. However after initial learning period, the human user can perform very complicated optimization tasks, by using the relative height and direction of movement of the sensitivity bars. Two very simple examples will be given to illustrate how *DynHist* can be used for optimization. Consider a model that represent on-design cycle of a two spool unmixed turbofan, where maximum overall efficiency is the optimum condition.

- Example 1. To find the optimum with respect to one independent variable: What is the optimum bypass ratio, all other variables remaining the same? Here the user turns the bypass dial, in the direction that is reducing the sensitivity bar (which designates sensitivity of overall efficiency with respect to bypass ratio) to zero. zero sensitivity and displayed value direction of overall efficiency bar indicate that the required optimum is found
- Example 2. To find the influence of one independent variable on the optimum value of another independent variable: What is the value of bypass ratio which makes the current value of fan pressure ratio optimum? Here user again turns the bypass dial until the sensitivity of overall efficiency goes to zero.

After exploring the model with *DynHist*, for say ten minutes, the user has gained a lot of experience and knowledge about the model and he can attempt to find say a global optimum. The direction in which user will search will be “*considered*” (based on the gained experience about the model) and not necessarily random or haphazard.

2.2.2 Dynamic Carpets “DynCarp”

As the number of independent variables increase, it can become somewhat difficult to make sense of all the moving histograms and all the changing directions. The *DynCarp* program was then developed to aid the optimization and better understanding of a multidimensional model. The basic principles are exactly the same as before (e.g. same

model source code is used as in *DynHist*), except that this time a carpet plot moves through the design space and changes its shape, as the values of independent variables are changed by turning dials.

Each carpet plot is a three dimensional surface, displaying two independent parameters and two dependent parameters. One of the independent parameters is also used as the third dimension to give a 3D surface and better visualization.

All the independent and dependent variables are defined in the model. The choice of which variables to show in the carpet is made graphically at run time. Both independent and dependent variable ranges can also be changed at run time by turning the dials.

Fig. 12 to Fig. 13 shows *DynCarp* in action, for an On-Design model of a separate exhaust turbofan. The value of any independent variable defined in the engine model can be continuously changed, e.g. the mach number, and the change in position and shape of carpet is observed in milliseconds.

DynCarp is a powerful aid to the human designer for understanding complex model behaviour. E.g. the following complex conclusion about the example engine can be drawn in several minutes, just by turning

the mach dial.

1. Overall efficiency levels increase as flight mach number increases
2. Specific thrust decreases as flight mach number increases
3. at low mach number, optimum combinations of fan pressure ratio and bypass ratio are, relatively high bypass, medium fan pressure ratios
4. at high mach number, optimum bypass ratio is very low, and optimum not very sensitive to fan pressure ratio
5. At high flight mach numbers, combination of high bypass ratios and high fan pressure ratios result in infeasible cycles

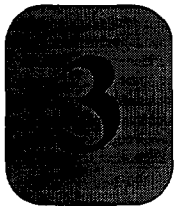
The above conclusions may seem simple, but *DynCarp* allows variation of any independent variable, e.g. component efficiencies or TET effortlessly and see for what conditions the above conclusions hold. This is consistent with the aim of the *Natural Design Cycle* to aid the human designer gain new experiences and reach new conclusions quickly rather than using a set of previously developed or reached conclusions, i.e. as opposed to knowledge based approaches. In this way the quality of design depends on the quality of the human designer.

2.3 Graphical User Interface GUI

The role of the *GUI* in the *Natural Design Cycle* and in *CAGED* has been sufficiently explained in the previous sections. Most readers will already know what a *GUI* is, because it is a hot topic at the moment and most current software employs it in one way or another. The *GUI* is basically very pleasant to use, but not necessarily to read about. The *GUI* in *CAGED* was required to be very efficient (i.e. minimum overhead to allow processing time for high speed design loop) and was therefore purpose written, independent of any other system. It only uses the *Graphics Library* and the *IRIX* (i.e. *UNIX* on *SGI Iris* workstations) operating system which automatically come with any *SGI Iris* work station. The current version of *GUI* makes the following possible,

1. all system functions are visually driven, by using mouse and screen buttons, i.e. post-processing, model definition, high speed are all effortlessly available.
2. user can define any engine configuration by picking standard components with mouse and visually defining the type of interaction between these components, e.g. aerodynamic link, mechanical link, artificial link etc.
3. the visual network can then be used for any parameter get/set operation
4. can recall previously defined models for modification and or inclusion within current model
5. allows recursive inclusion of models i.e. a model can include a model which includes a model which include a model.....
6. once an engine configuration is defined, then it is not saved in a complicated data file showing which components are at which stations and to which components they are connected to. Instead only the *User Interaction* which defined the model in the first place is saved. The user interaction is basically through the mouse and keyboard. Therefore a model file in *CAGED* is basically a data file with relevant mouse and keyboard interaction. Then every time a previously defined model is included, this previous user interaction is fed into the *GUI* and the system reacts precisely the same it did when the model was first defined. This makes model files extremely simple, compact and high speed to regenerate.

There is more discussion on the role of the *GUI* and its relevance to *NDC* in section 8.2. In addition, section 8.3.1 discusses the relevance of the *GUI* as a tool for engine analysis or as a general systems tool.



Brief Review of Past and Current Activities in Computer Aided Engine Design / Analysis

Early engine analysis programs were primarily developed for particular engine types of fixed configuration. *Customer Decks*, for example, are programs generated by engine manufacturers for particular engine models, accurately calculating engine thermodynamics, steady state as well as transient, some times together with sizing and weight calculations. These are still very much in use today by aircraft engine manufacturers studying feasibility and integration of a particular engine model into the airframe. This is consistent with the traditional approach of designing the engine and aircraft separately followed by integration.

As early as mid-sixties, the need for analysis of *General Engine Configurations* in the *Conceptual/Preliminary Design* domain was recognized, because,

1. The number and variation of engine concepts was growing and it was not efficient to develop and *validate* computer code for each concept separately
2. In conceptual/preliminary design, the designer wants to observe the effect of many configurational changes to his model.

Examples of early attempts at general analysis are, *SMOTE 1967*, *NASA GENENG* (GENeralized-ENGINE^[31]), *NASA DYNGEN* (GENENG + transient analysis)^[56].

The above approach to generalization was to use the analysis of a fixed complicated configuration of a gas turbine (3 spool-3 stream turbofan with afterburner and cooling) to analyse sub-derivatives of that configuration (e.g. a turbofan with zero bypass becomes a turbojet). This approach is not yet ideal for preliminary design because it is impossible to construct a super-complicated engine configuration of an engine such that all engines of interest are sub-derivatives of it. For example, suppose that the design

engineer is content with analysing a certain type of turbofan engine only. At some point it may be necessary say to find the effect of position of cooling air bleed in the high pressure compressor, or alternatively to study the effect of the addition of an exotic device. But if the engine cycle is fixed, these simple variations cannot be handled.

Later *NNEP* (Navy Nasa Engine Program)^[16], made it possible to connect a set of standard components in practically any configuration. This program was then enhanced to obtain *Conceptual/Preliminary Design* Program (as opposed to just thermodynamic analysis of cycles) as follows,

- *NNEPEQ*^[17]. This is the chemical equilibrium version of *NNEP* (i.e. *NNEP* + *CET86*^[25] to gain accuracy for where dissociation effects and various fuels need to be considered.
- Three programs^[9] (from *General Electric*) which calculate the component maps of compressors (*CMGEN*), turbines (*PART*) and fans (*MODFAN*) based on design variables, i.e. not a simple scaling of a given map.
- *Boeing WATE* (Weight Analysis of Turbine Engines)^[46]. Sizes each component and the mechanical layout of engine, according to the most critical conditions (i.e. maximum temperatures, pressures, shaft speeds etc.) reached in the likely flight envelope of the engine. Based on the sizing of each component, engine weight is then calculated to within 90% to 95% accuracy.
- *NEPAS* - recent addition of *GUI* enables the engine configuration to be defined graphically.

High Speed Engine Models were first analogue or hybrid models (e.g. *HYDES*^[60]) due to relatively low speed of digital computers at the time, used mostly for dynamic simulations and control studies of a particular cycles. Accurate high speed engine models are so desirable, that at *NAL*, ultra high speed computers (i.e. computer hardware specially designed for this job)^[59] are used for digital steady-state/dynamic simulation as well as design of gas-turbine engines.

Another line of activity in recent years, has been the application of artificial intelligence and object orientation to engine design (e.g. *ENGINEOUS*^[61] from *General Electric*). Among the reported advantages are,

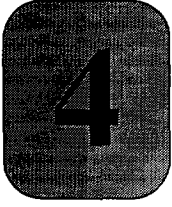
- Shorten the product design and development times by applying validated design rules resulting from many years of past experience. (i.e. *Knowledge Base* approach)

-
-
- Relative ease of integration of large number existing analysis routines and applying a combination of numerical, symbolic and other optimization techniques, due to *Object Orientation*.

(see references [1] [38] [32] for more information on the *Engineous* effort).

Despite the progress in computers and computerized engine design/analysis programs and systems, real engine design tends to be ruled by experienced human designers, and by trial and error, or “tweaking” engine hardware until the engine design is satisfactory.

However, researchers are already planning ahead for the time when orders are magnitude bigger and better computers are coupled with sophisticated multi-disciplinary analysis codes. Reference^[43] envisions a futuristic engine design system *NPSS* (Numerical Propulsion System Simulator). *NPSS* aims at all levels of the design of complete propulsion systems such that there is only a single hardware build-test prior to certification.



Aerothermodynamic Design Approach

In *CAGED* the engine configuration is defined by the *HD* with the aid of the *GUI* (See section 2.3) by graphically picking standard components and graphically defining all interactions (mass, work, heat transfer or artificial relations) between the selected components. Once an engine configuration is defined, *CAGED* is responsible for generating *On- and Off-Design* thermodynamic and *Transient* engine models, which are *instantaneous* in order to be suitable for employment in the *Natural Design Cycle*. “*Instantaneous*” means order of milliseconds execution time on *current* workstations, (e.g. equivalent graphical and processing capabilities of a 36 MHz SGI Iris Indigo). While traditional and current engine modelling practices can meet this time requirement using high speed mainframes, they are classified in this report as too sluggish. In *CAGED* high speed was achieved by application of good software practices, conceptual changes to modelling and analytical short-cuts.

In this chapter, first the *Universal Component* (the basic engine unit in *CAGED*) is discussed. This is the underlying reason behind the strength of *CAGED* engine models, because it can represent all components. Further, going from *On-Design* to *Off-Design* to *Mechanical Design* to *Variable Geometry* requires no change in this basic unit.

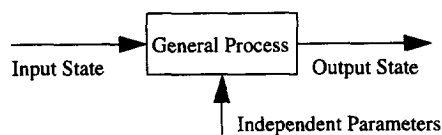
Then the *Sequential Network Logic*, its role in generating high speed On-Design model, and the structure of the final generated source code are explained.

In addition, *Direct* (i.e. closed form non-iterative to save time) *Analytical Solutions* to *Off-Design* thermodynamic calculation of a *Generic Spool* are derived, and their implementation into the engine *Off-Design* model is discussed together with the advantages that they can offer.

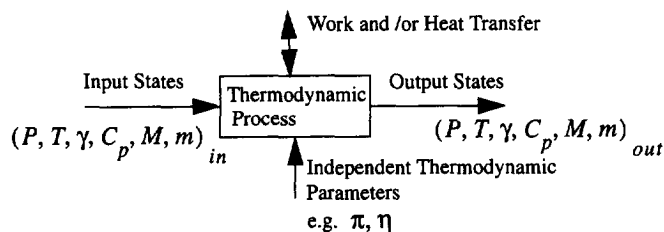
4.1 Engine Configuration Definition

4.1.1 The Universal Component

In order to arrive at a general method of component representation, the definition of a general process will be discussed, a black box that gives an output state depending on the input state and the independent parameters of the box. For example in a water valve the output flow is determined from the input flow and the valve position as the control variable;



Similarly a gas turbine component involves a *Thermodynamic Process* i.e. a change of the thermodynamic state of the fluid (pressure, temperature etc.) across the component, depending on the value of the independent thermodynamic parameters. For example, the thermodynamic process in a compressor (i.e. adiabatic compression) can be represented by an ideal isentropic process together with an isentropic efficiency and pressure ratio, as shown below.



Although the thermodynamic process is essential, it is by no means sufficient to represent all that happens in a component, because the independent thermodynamic parameters of a thermodynamic process are not allowed to take on any values freely,

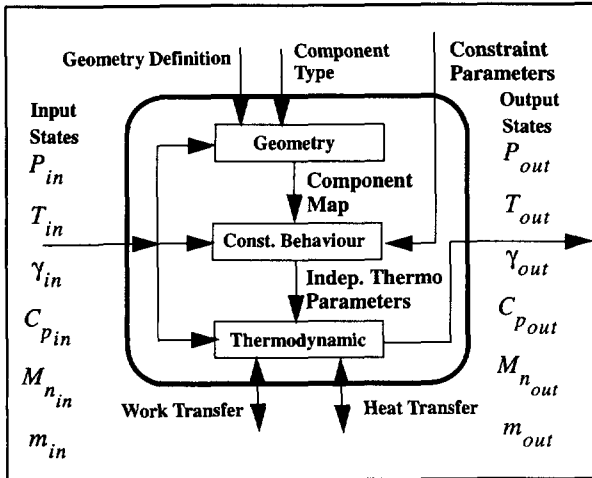
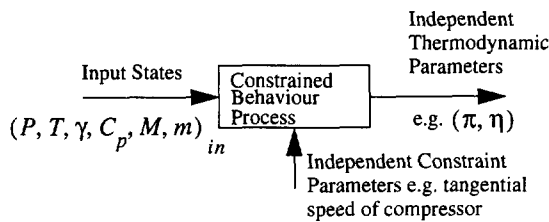


FIG. 15 The Basic Engine Component.
An engine is aerothermodynamically represented by an information network of such components.

but are in fact constrained by certain physical phenomena. For example, the values of compressor pressure ratio and efficiency, depend on the rotational speed that the compressor is allowed to operate at and the inlet conditions.

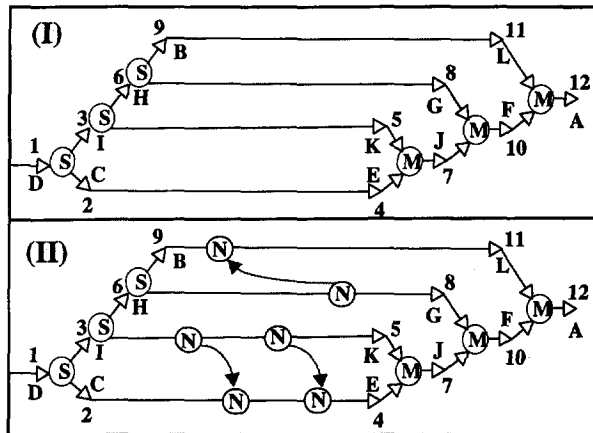
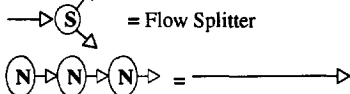
The constrained behaviour of components can be represented by the “Constrained Behaviour Process” as shown below. The purpose of this process is to give values of the parameters needed in the thermodynamic process, dependant on input conditions and values of constraint parameters,



For a given value of the independent constraint parameter (e.g speed), the constrained behaviour process above will give the thermodynamic independent parameters needed for the thermodynamic process. The thermodynamic process is then used to give the final thermodynamic conditions at output. For the compressor example, the constrained

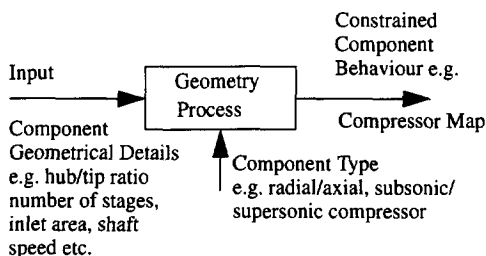
FIG. 16 (I): Example Information Network where information is flowing downstream. *Sequential Network Logic* puts any general network with incorrect sequence (e.g. A,B,C,D....) into correct sequence (1,2,3,4....). (II): (I) + interaction between normal components typical in a real engine network.

A,B,C.... = original station numbers
1,2,3..... = ordered station number after application of *SNL*



behaviour process would be reading a compressor map and calculating compressor pressure ratio for a given values of shaft speed and thermodynamic inlet conditions.

One aspect remains to be solved, namely how to determine the constrained physical behaviour of the component. This behaviour is clearly dependant on the type of component and its geometrical description, determination of which becomes the responsibility of the *Geometry Process*. This process determines the constrained behaviour of the component given geometrical details of the component as shown below,



To summarize, three basic processes are necessary to represent the basic component of propulsion systems,

1. geometry process,
2. constrained behaviour process,
3. thermodynamic process.

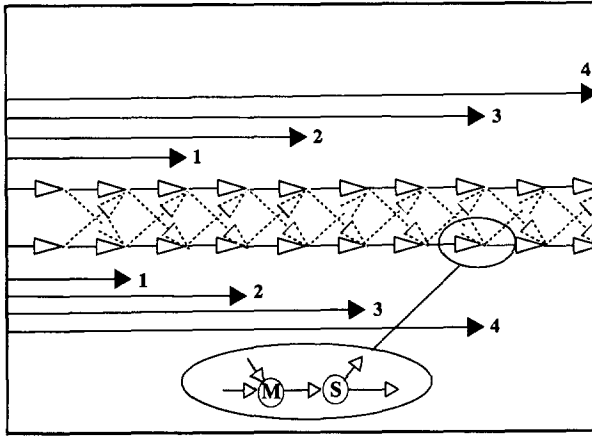


FIG. 17 Relation between the required number of calculation runs through the network and the number of unknown mixers in two parallel streams. Solid arrows indicate the extent to which the *Sequential Network* has solved the unknowns after each run. Dotted arrows indicate unknown flow of information between N components.

The *geometry* process determines the constrained component behaviour (component map) once the geometrical details and type of component are known. The *constrained behaviour* process, in turn, determines the independent thermodynamic process parameters once the values of input conditions and constraint parameters are given. Finally, the *thermodynamic* process gives the thermodynamic properties of the working fluid at the output end of the component.

A general representation of the basic component is a combination of three processes as shown in Fig. 15.

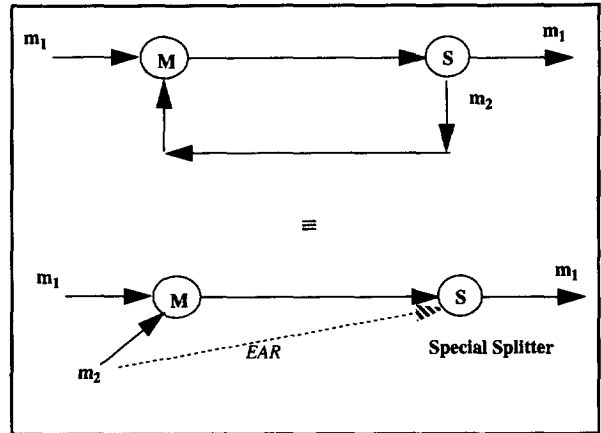
4.1.2 Interaction between Components

There are three basic types of thermodynamic interaction between components namely,

1. mass transfer,
2. work transfer,
3. heat transfer.

In a practical engine design system the above three are not enough and additional interactions are required which are here classified as *Externally Applied Relations* (EAR). EARs make it possible to artificially relate any parameter of one component (thermodynamic or mechanical) to parameters of other components or to other parameters of the same component. Care must be taken however to avoid reverse flow of information so that the *Serial Network Logic* will not fail (See section 4.2.1). One example for necessity of EARs is the convergent exhaust nozzle. In order to decide whether it is choked or not, it is necessary to know the output conditions of the nozzle, which we are trying to calculate. When the SNL calculation has marched through the network and arrives at the exhaust nozzle, an EAR from the mass source (i.e. the origin

FIG. 18 Replacing feed back (reverse flow of information) with forward flow equivalent.



of working fluid, the atmosphere) to this component sets the pressure ratio for this component after deciding whether it is choked or not.

Suppose that during an on-design run, we want to introduce correlations between parameters that are not well defined but definitely validated, although the designer must remain aware of the physical implications of such a relationship. An *EAR* from the compressor to the turbine will make this possible.

4.2 On Design Method

4.2.1 Source Code Generation for On-Design Performance of Engine Models & “Serial Network Logic”

Propulsion systems can be modelled as a system made up of a network of components. *Network Logic* is the method by which calculation marches through the network. *Parallel Network Logic* is where the state of the system is determined at all stations (or nodes) simultaneously. In a thermodynamic network, this is done by setting up and solving a set of simultaneous equations. While this practise is very general, i.e. can handle almost any network, it is very inefficient regarding calculation effort, and therefore in no way applicable to *Natural Design*. This section describes a *Sequential Network Logic* which can solve any general network of components, as long as the flow of information is downstream. Backward flow of information can also be represented by forward flow of information as will be discussed. The beauty of sequential network logic is that it is done only once and can be taken out of the final engine model if the engine configuration is fixed during a design run.

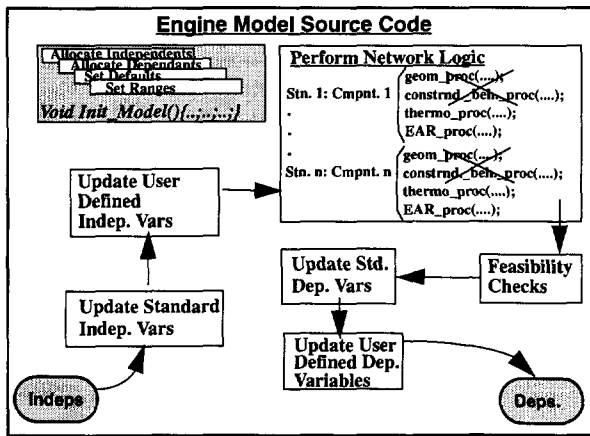


FIG. 19 Structure of Engine Models generated by CAGED. The unshaded blocks represent the function `void gt(...)` which is the high speed function continuously being called. The function `void init_model(...)` is called only once every time a new configuration or a new set of user defined variables is required. If On-Design Thermodynamics is required only, then the geometry and constrained behaviour process for each component can be omitted as shown.

In a general information network, there are basically three types of components. *Splitters*, *Mixers* and *Normal* components (Fig. 16 (I)). *Normal* components output information based on a single stream of input. *Splitters* split the incoming information stream into two and *Mixers* perform the opposite.

For a general information network, where information is flowing down stream, *Sequential Network Logic* states,

- The conditions at each node is found correctly by marching through each stream of information from start to end.

In example case (Fig. 16 (I)) this means the following sequence of marching will fully solve the network,

- DCEJFA, DIKJFA, DIHGFA, DIHBLA

After each sequence the conditions at each node are stored and used for the next sequence and so on. The above sequence is wasteful because there is a lot of unnecessary marching. E.g. in the first sequence it is wasteful to march further to *J* because conditions at *K* are not yet correctly known, and any marching after mixer is not solving any more unknowns. The best way through the example network is to march in the sequence 123456789.10.11.12 as shown, much less effort than the above sequencing.

The *Sequential Network Logic* implemented in CAGED transforms any unordered forward flowing information network of *M*, *N* and *S* components (e.g. Fig. 16 (I) with letters) into an ordered network (e.g. Fig. 16 (I) with numbers), no matter how complicated the network is. The basic method is as follows,

- If output of a S is input to another S , then the latter is a sub S and the former the mother S . There is no limit on recursive continuation of this relation. Every time a S is encountered, the current S becomes the mother S and the new S becomes the current S .
- Take any stream and continue until a M . Go back to current S (if any) and start again on the secondary stream until M . Now both streams have been processed and can go through M until another M .
- Go back to mother S and continue.

In the thermodynamic sense, an engine or a fluid network can be represented by such an information network. In other words, the flow of fluid is represented as flow of information. In line components, fluid bypass and fluid mixers are represented by N , S and M respectively. Typically in engines, there is flow of informationⁱ (separate to flow of fluid) between N 's (Fig. 16 (II)). E.g. compressor driven by turbine, work transfer, or some artificial relation between components. This is handled adequately by considering the N emitting information as S and the N receiving information as M .

Once the quickest possible sequence of marching through network is found, it remains constant if the configuration of engine and flow of information is constant. Further the types of components (e.g. compressor, turbine, burner etc.) that make up the network and the necessary routines to be called for each component, are known to *CAGED*. This means that *CAGED* is able to generate a *ANSI C* source code which contains the solution to the engine thermodynamic network. The source code is made up of calls to subroutines in correct sequence. No execution time is wasted in deciding where to go next in the network.

If there is unknownⁱⁱ flow of information between components, then the *Sequential Network Logic* becomes blind to the corresponding splitting and mixing of information, implying that a single run is not adequate to solve the network. The cause of the unknown is the *mixer* since both information input channels must be known to calculate output. Fig. 17 shows two known parallel network streams with unknown *EARs* acting between them. It is clear to see that after the first run of *Sequential Network Logic*, only

-
- i. The flow of information between components other than fluid mass is called an *EAR* (externally applied relation)
 - ii. flow of information is "unknown" if the actual emitting and/or receiving component is not known in advance and becomes clear only at run time.

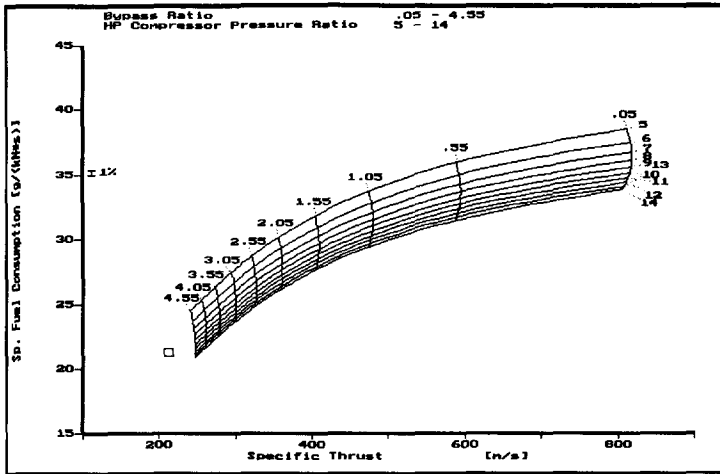


FIG. 20 a) Design Point
Performance of an
unmixed turbofan.
Design point:
altitude = 10668 m,
Mach = 0.8, TET =
1800°K, fan
pressure ratio = 2.0.

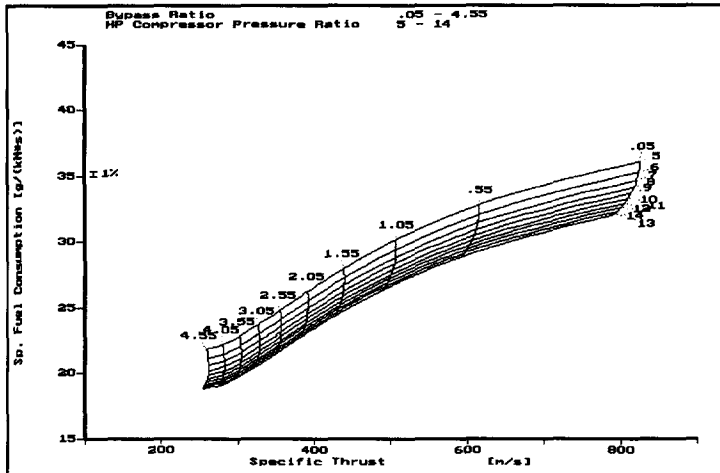


FIG. 20 b) fan pressure ratio =
3.

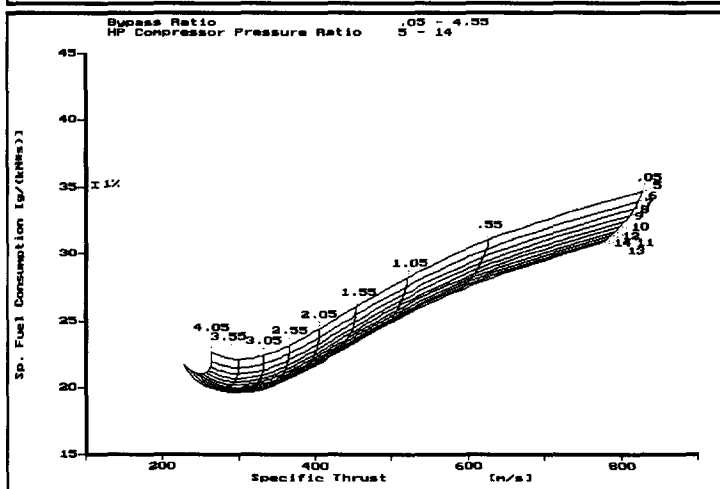


FIG. 20 c) hp-fan pressure
ratio = 4.

FIG. 21 a) Design point
performance of
unmixed turbofan.
Design point:
TET=1800, HP
pressure ratio= 7.0,
Flight Mach=0.8.

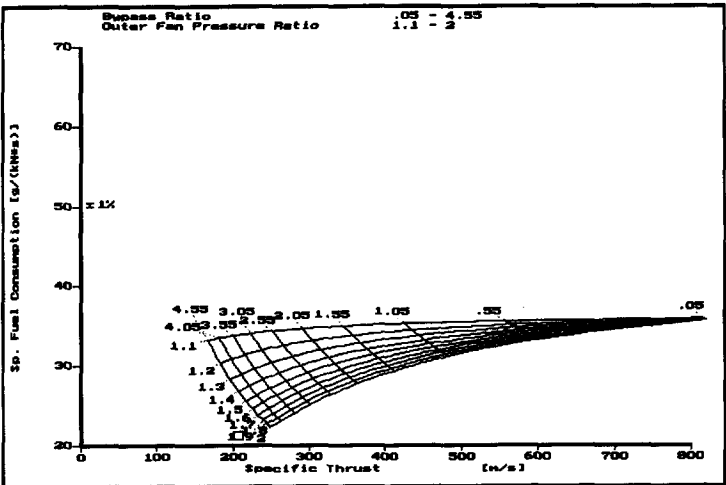


FIG. 21 b) Flight Mach=1.75

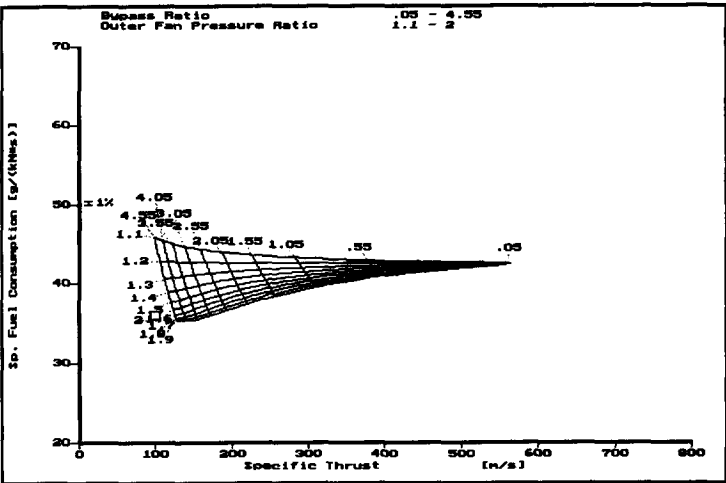
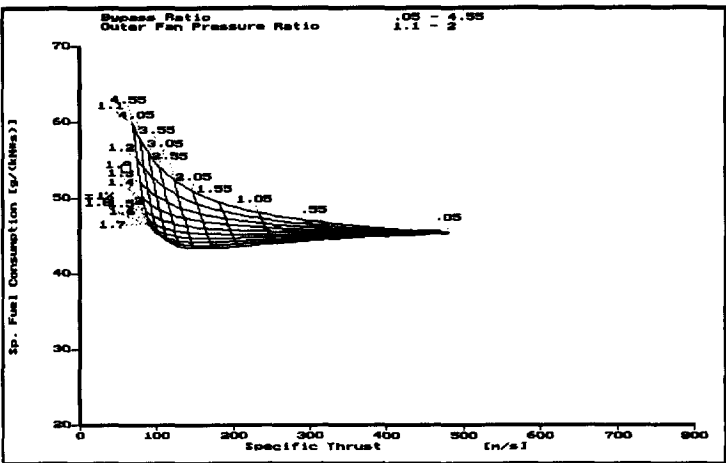


FIG. 21 c) Flight Mach=2.0.



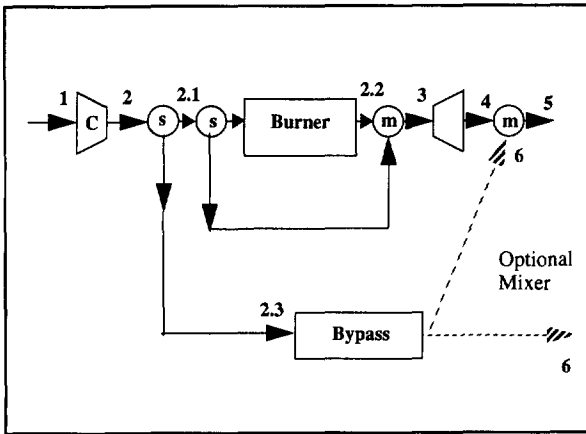


FIG. 22 The Generic Spool. The components shown here represent the “*Genric Spool*” altogether, and not just the compressor/turbine pair. The “*Burner*” can be a real burner or an included generic spool as shown in Fig. 23 .

part of the two streams has been solved, due to unknown mixing. The relation between the number of *SNL* runs and the number of included unknown mixers on a stream is,

$$N_{Mix} = (N_{Run} - 1) \times 2 \quad (1)$$

An example of unknown flow of information is a variable cycle or a variable configuration engine, where the variation cannot be fixed or decided in advance and can be known only at run time of engine model depending on the state of the model at any given moment.

Most engines known to the author can be adequately represented by a forward flowing information network. Flow of information is backward if a piece of information collides with itself (e.g. fluid feed back).

Each reverse flow of information causes an unknown in the *Network Logic*. If this reverse flow cannot be modelled in some way by forward flow then it is still wasteful in most cases to feed the whole network into a parallel solver because the amount of reverse flow is relatively small in most engine concepts. One must then devise a *Semi-Parallel Network Logic* which solves *Sequentially* by default and performs parallel processing only where it is really needed. A simple example of reverse flow of information is a spool where information is going from turbine to compressor, i.e. work transfer. Stated in this way, this needs parallel processing. But if we considering negative work is transferred from compressor to turbine, then reverse flow is replaced by forward flow. The fluid feedback can be handled similarly. (Fig. 18). The feed back flow conditions are specified and an *EAR* takes this information to a special splitter, which evaluates whether the feed back conditions are feasible and evaluates the output conditions for this feed back condition.

Once the correct marching sequence through the network is determined, then it is relatively simple to generate the source code for the engine configuration at hand. Source code generation is performed interactively with the *HD* (i.e. via the *GUI*). *Fig. 19* shows the major components of the source code that is generated by *CAGED*. The function *Void Init_model(...,...)*; is responsible for setting up the model problem and fixes all the information that need to be generated only once for a given engine configuration and set of variables interesting to the *HD*. The function *Void gt(...,...)*; is the high speed link between the *Independent* and the *Dependent* variables. The user defined independent variables can be a simple selection of a components geometric/thermodynamic variable/property or a user defined combination thereof. Standard independent variables are flight mach number and pressure altitude. Standard dependent variables are specific thrust F_s , *SFC* etcetera. User defined dependent variables are any user defined combination of any property or variable of the engine model.

4.3 Cycle Selection and Sizing

The exact method of how to select and size the engine cycle varies between designers and authors. The current effort tries to supply the *HD* with the tools that he/she needs instead of fixing a particular methodology for selecting the cycle and sizing the engine. Suffice it to say that with the aid of high speed engine models, it is possible to move through the design space in real time. This helps the *HD* gain quick experience and reach complex conclusions about the engine model and aids in the selection of a design point for the engine. For example the time taken to move from *Fig. 20 a) to b) to c)* (and from *Fig. 21 a) to b) to c)*) is in the order of milliseconds on a *SGI 36 MHz Iris Indigo*.

Since the engine model includes any user defined dependant variables, the user can define any merit functions in order to aid him in the selection of cycle. Selecting the On-Design point for the engine and sizing it to deliver a given thrust (i.e. selecting a required mass flow rate, and calculating the flow path areas) is the embarkation point for the calculation of off-design performance.

4.4 Off-Design Analysis

Methods to calculate Off-Design performance engines are standard practise and discussed in sufficient detail in most basic or introductory textbooks^{[8] [28] [29]}.

Once the cycle is selected and the engine sized, then any new steady state operating condition, (e.g. change in properties of intake fluid caused by altitude and/or flight mach number or different throttle position etc.) is found by iteration of variables until the conservation of mass, energy and momentum and the second law of thermodynamics are satisfied.

Modern and accurate thermodynamic analysis programs (e.g. *NNEPEQ*) set up N simultaneous equations (from mass, energy, momentum and 2nd law, thermodynamic processes) in N unknowns (fluid properties at each engine station) for a general thermodynamic network. The problem is then fed into a general purpose solver (*NNEPEQ* uses *Multivariate Newton Raphson*) which will find the solution by iteration. This is called the *Parallel* method in this report.

Since the number of iteration variables is large, and the equations are not linear in general, the *Parallel* method tends to be very sluggish and not suitable for *Natural Design*. Drastic improvement in speed is possible by applying numerical tricks to the *Multivariate Newton Raphson* (See section 6.6) but is still not enough if the engine network is moderately complicated with relatively large number of components. Another general purpose solver/optimizer is the *Non-Random Adaptive Grid* which is advantageous (i.e. less calculation effort) for highly dimensional and difficult functions (See section 6.1).

The *Parallel* approach is very general and can be very accurate if suitable component maps are used. However the *Parallel* approach is very sluggish, due to the iterative nature of solution and the large number of unknowns, and hence not consistent with the *Natural Design* philosophy of *CAGED*.

The need for high speed off-design methods should not be underestimated. An instantaneous Off-Design method would make the following possible,

1. *Integrated Aerothermodynamic and Mechanical Natural Design*. Past engine design activity has lead to a set of mechanical design rules for each engine component and rules on assembly and interaction between various components which are consistent with conceptual and preliminary design level accuracies (e.g. the *Boeing WATE*^[46] code). These rules are based on critical conditions (e.g. maximum temperature, wheel speeds etc.) that occur as the engine flies through its flight envelope. Each time the on-design variables changeⁱ, the off-design performance of engine changes and so will the critical conditions reached

i. i.e. each time the engine on-design point changes.

for every component. If the off-design calculation method was so fast such that the critical off-design points of the flight envelope are calculated instantaneously every time the on-design point is changed, then it is possible to see the effect of on-design thermodynamic variables on the mechanical design of the engine in *Real Time*.

2. *Integrated Aerothermodynamic and Mission Analysis*. Engines are generally designed for a range of operating conditions rather than a single operating point. If one operating point is considered as the On-Design point, then the rest of the points become Off-Design points. An instantaneous Off-Design methodology would make it possible to change the On-Design thermodynamic variables and observe engine behaviour at all the other mission points where the engine is expected to operate, in *Real Time*. Thus designing for multiple operating points rather than a single On-Design point.

Fast analytical methods for Off-Design calculations fall into three categories,

1. *Global Analysis*. These are (semi-)analytical relations derived from the thrust equation and dimensional analysis. The engine is treated as black box and the internal configuration and details are not considered.
2. *Choked Turbines or Nozzles*. Direct solution to the off-design equations by the assumption that the internal spools are operating choked. The outermost spool is then either assumed choked (giving a direct solution) or coupled to exhaust nozzle characteristics (giving iterative solution with one iteration variable).
3. *Differential or Sensitivity Analysis*. Here slowly varying functions vanish by themselves. It is therefore not necessary to assume choked conditions at inlet to or outlet from the turbines or nozzles. The assumption of *Well Designed Well Matched* spool together with minimal information on the type of first turbine stage (e.g. impulse or 50% reaction etc.) then leads to direct analytical solution for the *Generic Spool* regardless of how restricted the flow is downstream of the spool.

The *Global Analysis* is typically used in aircraft preliminary design projects. They are actually not very useful in engine design since no internal engine details are included. A list of global and choked turbine off-design relations^{[40] [63]}, developed by other authors, is provided in the appendix for reference.

The closed form solutions derived by the author for the steady state off-design performance of a generic spool falls into the third category above and is explained in the following sections.

4.4.1 Direct Non-Iterative Solution For a Generic Spool

Once the engine thermodynamic cycle is fixed and sized, then the design point thermodynamic parameters of each component are fixed. The components that do not require external information other than that contained in the input, represent absolutely no problemⁱ to the *Off-Design* solution. For these components all the information is available so that the new output properties can be calculated simply from the new input properties. These would be the N components in the information network drawn up previously.

Most engines can be considered to be a network of forward flowing information, in the *On-Design* context as discussed above. Unfortunately, in the *Off-Design* domain, the same engine networks have backward flow of information. For example, for a compressor driven by a turbine, the new operating point of the compressor depends on how much power is transferred to it by the corresponding turbine for which the operating point is not known, since it depends on the compressor outlet flow. A cycle of trial and error on the compressor operating point finds the conditions where the turbine and the compressor are matched.

Typically, the only backward flow of information is caused by the turbine driving the compressor as discussed. For this reason, an attempt is made in this section to delete the reverse information flow (i.e. the work transfer from a turbine to a compressor) in a generic spool. Success of this then enables the *SNL* to be used for *Off-Design* as well as for *On-Design* calculations.

The following sections derive the non-iterative closed form solution to the off-design equations of a generic spool. For the benefit of the reader, first a solution is derived for the generic spool with *restricted* outlet because of its relative simplicity, applicable to high and intermediate pressure spools. This is followed by finding the solution for a generic spool whose outlet is *not restricted*, i.e. one whose outlet is matched to the characteristics of an expanding nozzle placed downstream of the spool, applicable to the low pressure spool. Furthermore a third direct solution is derived which does away with the question of restriction entirely, by requiring simple information about the type of turbine stage and making a key assumption that the compressor and the turbine are *well designed and well matched*.

i. i.e. those components that do not cause reverse flow of information, for example output conditions of a duct can be determined from known input conditions, i.e. there is no information required that comes from some other component downstream in the network.

Consider the generic spool as defined in *Fig. 22* . The basic relations that describe the steady state operation of the generic spool are as follows,

Mass Balance 1 to 4 gives:

$$\pi_s = \frac{A_1 \mu_1}{A_4 \mu_4} m_{rat} \sqrt{\tau_s} \quad (2)$$

$$\tau_s = \left\{ \frac{\varnothing^{a_t}}{\pi_b \tau_c^{a_c}} \times \frac{A_1 \mu_1}{A_4 \mu_4} \right\}^{\frac{2}{2a_t - 1}} \quad (3)$$

Power Balance Compressor-Turbine:

$$\tau_s = \varnothing - \frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \quad (4)$$

$$\tau_t = 1 - \frac{\tau_c - 1}{C_{p_{rat}} m_{rat} \varnothing} \quad (5)$$

Mass balance 1 to 3:

$$\varnothing = \left\{ \frac{\pi_b \tau_c^{a_c} A_3 \mu_3}{m_{rat} A_1 \mu_1} \right\}^2 \quad (6)$$

Where μ, π and τ denote the non-dimensional mass flow parameter, stagnation pressure ratio and stagnation temperature ratio respectively. Subscripts $s, c, b, t, 1$ to 6 and rat represent the spool, compressor, burner, turbine, station numbers and the "ratio" of a quantity at station 3 to station 1 respectively. For example $\tau_s = T_{t4}/T_{t1}$, $\pi_c = p_{t2}/p_{t1}$ and $m_{rat} = m_3/m_1$.

Furthermore the compressor and turbine pressure ratios are assumed to be polytropically related to the corresponding temperature ratios as follows,

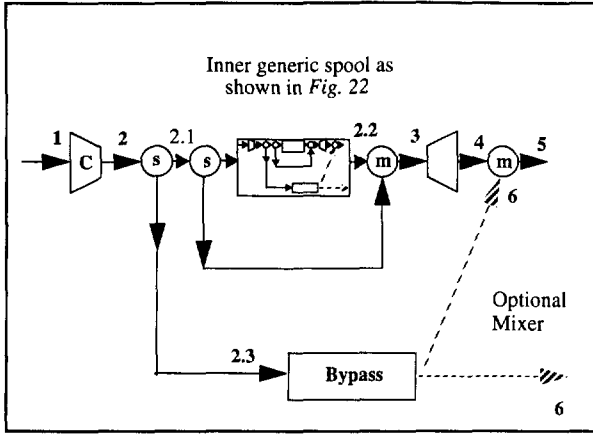


FIG. 23 Recursive inclusion of Generic Spools. Calculation proceeds from the innermost spool to the outermost, where the $\nabla \pi_s$ term for each spool provides the $\nabla \pi_b$ term for the enclosing spool.

$$\pi_c = \tau_c^{a_c}, a_c = \frac{\gamma_c \eta_c}{\gamma_c - 1} \quad (7)$$

$$\pi_t = \tau_t^{a_t}, a_t = \frac{\gamma_t}{\eta_t (\gamma_t - 1)} \quad (8)$$

In the following text the dimensionless derivative operator (or log derivative),

$$\nabla_\mu X = \frac{\partial X}{X} / \frac{\partial \mu}{\mu} = \frac{\mu}{X} \frac{\partial X}{\partial \mu} = \frac{\partial \log X}{\partial \log \mu} \quad (9)$$

is always w.r.t. the mass flow parameter μ , where,

$$\mu = \frac{m \sqrt{RT_t}}{Ap_t} = \sqrt{\frac{\gamma}{R}} M \left[1 + \frac{\gamma - 1}{2} M^2 \right]^{-\frac{(\gamma + 1)}{2(\gamma - 1)}} \quad (10)$$

If the derivative is w.r.t. mass flow parameter at the compressor inlet (i.e. at station 1,

$\mu_1 = \frac{m_1 \sqrt{RT_{t1}}}{A_1 p_{t1}}$) then the subscript μ_1 is stripped, i.e. $\nabla_{\mu_1} X = \nabla X$ by definition.

Mathematical rules concerning ∇ operator are,

$$\begin{aligned}
 \nabla(YZ) &= \nabla Y + \nabla Z; \quad \nabla(Y/Z) = \nabla Y - \nabla Z \\
 \nabla(aZ) &= \nabla Z; \quad \nabla(Y+Z) = \frac{1}{Y+Z} (Y\nabla Y + Z\nabla Z) \\
 \nabla(Y^Z) &= Z\nabla Y + \nabla Z \log(Y^Z)
 \end{aligned} \tag{11}$$

where a is a constant, and \log represents natural logarithm.

4.4.1.1 Spool With Restricted Outlet

A spool is restricted when the $\nabla\mu_4$ term in eq. 19 is negligibly small compared to other terms. This is a robust mathematical definition for the “restricted” condition. In practice the high and intermediate spools can be considered as restricted. Figure 58 on page 183 compares the magnitude of the various terms that appear in eq. 19 for the high pressure spool of a two spool turbofan test case that will be used for validation in section 7.2 on page 174. The figure shows that the $\nabla\mu_4$ term is very much less significant than the other terms in eq. 19. Here the operating point of the turbine can be considered to be “restricted” by the flow at outlet. Note that “restricted” does not necessarily imply choked flow (i.e. Mach = 1.0) at turbine outlet. The only implication is that the relative variation in Mach number is small at the turbine outlet. However, if the turbine outlet is choked within a part of its operating range, then it is definitely restricted in that range.

Applying the ∇ operation to eq. 6 for negligible variation in A_3/A_1 ,

$$\nabla\phi = 2(-1 + \nabla\pi_c + \nabla\pi_b - \nabla m_{rat} + \cancel{\nabla\mu_3}) \tag{12}$$

The $\nabla\mu_3$ term is neglected because in practice it is very small compared to the other terms in the above equation whether or not the outlet is restricted (See Fig. 57 and Fig. 59 for illustration of this).

Application of ∇ to eq. 5 gives,

$$\begin{aligned}
\nabla \tau_t &= -\frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \left(\frac{\tau_c}{\tau_c - 1} \nabla \tau_c - \nabla \varnothing - \nabla m_{rat} \right) \\
&= -\frac{2(\tau_c - 1)}{C_{p_{rat}} m_{rat} \tau_s} \left(\frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \nabla \pi_c - (-1 + \nabla \pi_c + \nabla \pi_b - \nabla m_{rat}) - \frac{\nabla m_{rat}}{2} \right) \quad (13) \\
&= -\frac{2(\tau_c - 1)}{C_{p_{rat}} m_{rat} \tau_s} \left(\nabla \pi_c \left(\frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} - 1 \right) - \nabla \pi_b + \frac{\nabla m_{rat}}{2} + 1 \right)
\end{aligned}$$

Also,

$$\nabla \pi_t = a_t \nabla \tau_t \quad (14)$$

and rearranging eq. 4,

$$\frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} = \varnothing - \tau_s \quad (15)$$

Substituting for $(\nabla \pi_t$ from eq. 14) and $((\tau_c - 1)/C_{p_{rat}} m_{rat})$ from eq. 15 into eq. 13 gives,

$$\nabla \pi_t = \frac{2a_t(\varnothing - \tau_s)}{\tau_s} \left(\nabla \pi_c \left(1 - \frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \right) + \nabla \pi_b - \frac{\nabla m_{rat}}{2} - 1 \right) \quad (16)$$

Now $\pi_s = \pi_c \pi_b \pi_t$ so $\nabla \pi_s = \nabla \pi_c + \nabla \pi_b + \nabla \pi_t$, and substituting $\nabla \pi_t$ from eq. 16,

$$\begin{aligned}
\nabla \pi_s &= \nabla \pi_c \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} \left(1 - \frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \right) + 1 \right) + \nabla \pi_b \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} + 1 \right) \\
&\quad + \frac{2a_t(\varnothing - \tau_s)}{\tau_s} \left(-1 - \frac{\nabla m_{rat}}{2} \right) \quad (17)
\end{aligned}$$

Applying the ∇ operation to eq. 2,

$$\nabla \pi_s = 1 + \frac{\nabla \tau_s}{2} + \nabla m_{rat} - \dot{\nabla} \mu_4 \quad (18)$$

or

$$\frac{\nabla \tau_s}{2} = \nabla \pi_s - \nabla m_{rat} - 1 + \dot{\nabla} \mu_4 \quad (19)$$

Again the term $\nabla \mu_4$ has been neglected because this section treats a generic spool with restricted outlet. The non-restricted case is treated separately in section 4.4.1.4 on page 69.

Substituting for $\nabla \pi_s$ from eq. 17 into eq. 19 and isolating $(\nabla \tau_s)/2$ gives,

$$\begin{aligned} \frac{\nabla \tau_s}{2} = \nabla \pi_s - \nabla m_{rat} - 1 = \\ \nabla \pi_c \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} \left(1 - \frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \right) + 1 \right) + \nabla \pi_b \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} + 1 \right) + \\ + \nabla m_{rat} \left(-\frac{a_t(\varnothing - \tau_s)}{\tau_s} - 1 \right) + \left(-1 - \frac{2a_t(\varnothing - \tau_s)}{\tau_s} \right) \end{aligned} \quad (20)$$

Applying the ∇ operation to eq. 4 gives,

$$\begin{aligned} \frac{\nabla \tau_s}{2} = \frac{\mu_1}{2\tau_s} \frac{\partial}{\partial \mu_1} \left\{ \varnothing - \frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \right\} = \frac{\mu_1}{2\tau_s} \left\{ \frac{\partial \varnothing}{\partial \mu_1} - \frac{\partial}{\partial \mu_1} \left(\frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \right) \right\} \\ = \frac{1}{2\tau_s} \left(\varnothing \nabla \varnothing - \frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \nabla \left(\frac{\tau_c - 1}{C_{p_{rat}} m_{rat}} \right) \right) \end{aligned} \quad (21)$$

Substituting for $\nabla \varnothing$ from eq. 12:

$$\begin{aligned}
 \frac{\nabla \tau_s}{2} &= \frac{1}{\tau_s} \left(\emptyset (-1 + \nabla \pi_c + \nabla \pi_b - \nabla m_{rat} + \nabla \mu_3) - \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat}} \left(\frac{\tau_c}{\tau_c - 1} \nabla \tau_c - \nabla m_{rat} \right) \right) \\
 &= \frac{1}{\tau_s} \left\{ \nabla \pi_c \left(\emptyset - \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat}} \frac{\tau_c}{\tau_c - 1} \frac{1}{a_c} \right) + (\nabla \pi_b - 1) \emptyset + \nabla m_{rat} \left(-\emptyset + \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat}} \right) \right\} \quad (22)
 \end{aligned}$$

$$\boxed{
 \begin{aligned}
 \frac{\nabla \tau_s}{2} &= \frac{\nabla \pi_c}{\tau_s} \left(\emptyset - \frac{\tau_c}{2C_{p_{rat}} m_{rat} a_c} \right) + \frac{\emptyset}{\tau_s} (\nabla \pi_b - 1) \\
 &\quad + \frac{\nabla m_{rat}}{\tau_s} \left(-\emptyset + \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat}} \right)
 \end{aligned}
 } \quad (23)$$

Equating the two $\nabla \tau_s$ terms from eq. 23 & eq. 20 gives,

$$\begin{aligned}
 \nabla \pi_c \left(\frac{\emptyset - \tau_s}{\tau_s} \left(1 - \frac{\tau_c}{(\tau_c - 1) 2a_c} \right) (2a_t - 1) \right) &= \nabla \pi_b \left(\frac{\emptyset - \tau_s}{\tau_s} (1 - 2a_t) \right) \\
 + \nabla m_{rat} \left(\underbrace{-\frac{\emptyset}{\tau_s} + \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat} \tau_s} + 1 + a_t \frac{\emptyset - \tau_s}{\tau_s}}_{\frac{\emptyset - \tau_s}{\tau_s} \left(a_t - \frac{1}{2} \right)} \right) &+ \frac{\emptyset - \tau_s}{\tau_s} (2a_t - 1)
 \end{aligned} \quad (24)$$

Dividing through by $\frac{\emptyset - \tau_s}{\tau_s} (2a_t - 1)$ simplifies to,

$$\nabla \pi_c = \frac{1 - \nabla \pi_b + \frac{\nabla m_{rat}}{2}}{1 - \frac{\tau_c}{(\tau_c - 1) 2 a_c}} \quad (25)$$

The above relation gives the approximate slope of the equilibrium running line (*ERL*) in the compressor map at any given operating point of the spool. The $\nabla \mu_4$ term has been neglected in *eq. 18* and so it is applicable to cases where the flow is strongly restricted downstream of the spool (i.e. at station 4). The internal spools of a multi-spool engine satisfy this requirement in practice since the flow has to pass through several expansion components downstream before being released to the atmosphere. Any new operating point (not too far from the current operating point) can now be found because the gradient of the compressor pressure ratio is directly known at any given operating point. It can be observed that with this relation, the backward flow of information has been replaced by a forward flow of information, and the new operating point of the spool is directly known for any new input μ_1 to the spool. The $\nabla \pi_b$ and ∇m_{rat} are derived in the following sections.

Different levels of accuracy can be used with *eq. 25*. For example, the compressor efficiency (a_c term) can be assumed constant but may also be allowed to vary as a function of μ_1 , also the mass ratio term can be set equal to zero or allowed to vary as function of μ_1 depending on how much information is available.

The above analysis assumes zero shaft power off-take, and negligible ∇a_c , but a correction is provided in section 4.4.1.9 on page 83 for cases where these terms are significant. A constant mechanical efficiency of the spool could have been easily included in the compressor/turbine power balance equation (*eq. 4*) but would have disappeared from the final result, just like the $C_{p_{rat}}$ term.

The absence of some parameters in *eq. 18* is particularly interesting. The above suggests that turbine efficiency does not have any effect on the slope of the *ERL* and perhaps that is why the assumption of constant turbine efficiency does not lead to large errors in most off-design methods where the flow downstream of the spool is strongly restricted. Also the throttle position (represented by the ϕ term) seems to have no direct effect on the slope of the *ERL*.

4.4.1.2 Burner or Duct Pressure Loss Term

The stagnation pressure drop across the combustor is typically divided into two groups, the “hot” and “cold” loss. The former is the stagnation pressure drop as a result of heat addition and is not a strong function of μ_{in} . The latter is the result of internal drag of combustor which is a strong function of μ_{in} . Typically the hot loss of a combustor is several times smaller than its cold loss and is therefore neglected in the following derivation.

In one-dimensional flow, the pressure loss (i.e. cold loss) across any aerodynamic duct is^[39],

$$\pi = 1 - \frac{\mu_{in}^2 \Delta p_t}{2q} \quad (26)$$

where π is the stagnation pressure ratio, q is the dynamic pressure and Δp_t is the stagnation pressure loss term. The $\frac{\Delta p_t}{q}$ term is a non-dimensional measure of internal drag (or in some texts referred to as pressure loss factor or internal drag coefficient) and is assumed to be constant. The subscript “in” denotes entry station to component, which is the burner in this section.

Applying the ∇ operation to eq. 26,

$$\nabla_{\mu_{in}} \pi_b = 2 \left(\frac{\pi_b - 1}{\pi_b} \right) \quad (27)$$

It remains to find this gradient with respect to μ_1 and not μ_{in} , which comes from the mass balance between compressor inlet and burner inlet,

$$\begin{aligned}
\mu_{in} &\approx \mu_1 \frac{m_{2,1}}{m_1} \sqrt{\frac{T_{t2}}{T_{t1}}} \frac{P_{t1}}{P_{t2}} \frac{A_1}{A_2} \approx \mu_1 \frac{m_3}{m_1} \frac{\sqrt{\tau_c} A_1}{\pi_c A_2} \\
&\approx \mu_1 \frac{m_3}{m_1} \frac{\sqrt{\pi_c^{1/a_c}} A_1}{\pi_c A_2} \approx \mu_1 m_{rat} \frac{A_1}{A_2} \frac{1}{2a_c} - 1
\end{aligned} \tag{28}$$

Applying the ∇ operator,

$$\nabla \mu_{in} = 1 + \nabla m_{rat} + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \tag{29}$$

where duct areas have been assumed constant. In general, the non-dimensional gradient of variable X with respect to non-dimensional mass flow parameter at inlet to burner (or inner spool) can be transformed as follows,

$$\nabla_{\mu_1} X = \nabla_{\mu_{in}} X \times \left(1 + \nabla m_{rat} + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right) \tag{30}$$

In addition, the following relation is used to transform $\nabla_{\mu_3} X$ into $\nabla_{\mu_1} X$ where stations 1 and 3 are arbitrary station numbers.

$$\nabla_{\mu_1} X = \nabla_{\mu_3} X \times \left(1 + \nabla_{\mu_1} \left(\frac{m_3}{m_1} \right) - \nabla_{\mu_1} \left(\frac{P_{t3}}{P_{t1}} \right) + \frac{1}{2} \nabla_{\mu_1} \left(\frac{T_{t3}}{T_{t1}} \right) \right) \tag{31}$$

Using this transformation in eq. 27 then gives the non-dimensional gradient of burner pressure ratio with respect to mass flow parameter at inlet to compressor,

$$\nabla \pi_b = \nabla_{\mu_1} \pi_b = 2 \left(\frac{\pi_b - 1}{\pi_b} \right) \left(1 + \nabla m_{rat} + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right) \tag{32}$$

Substituting $\nabla \pi_c$ from eq. 32 into eq. 25, and solving for $\nabla \pi_c$ results in,

$$\nabla \pi_c = \frac{1 - 2 \left(\frac{\pi_b - 1}{\pi_b} \right) \left(1 + \nabla m_{rat} + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right) + \frac{\nabla m_{rat}}{2}}{1 - \frac{\tau_c}{(\tau_c - 1) 2a_c}} \quad (33)$$

which yields for $\nabla \pi_c$,

$$\therefore \nabla \pi_c = \frac{1 - J + \nabla m_{rat} \left(\frac{1}{2} - J \right)}{H + JK} \quad (34)$$

where,

$$\begin{cases} J = 2 \left(\frac{\pi_b - 1}{\pi_b} \right) \\ K = \frac{1}{2a_c} - 1 \\ H = 1 - \frac{\tau_c}{(\tau_c - 1) 2a_c} \end{cases} \quad (35)$$

The mass ratio term will be derived in the following sections.

4.4.1.3 Pressure Gradient Term of Downstream Components

In situations where $\nabla \mu_4$ cannot be ignored (e.g. for an lp-spool), it is necessary to consider the flow characteristics of downstream components. These downstream components are either a nozzle or a combination of components that can be considered as a general aerodynamic duct that expands the outlet flow of the spool into the atmosphere.

The corrected mass flow through an expanding duct is,

$$\mu_{in} = \frac{m \sqrt{RT_{in}}}{P_{t_{in}} A_{in}} = \frac{P_{out}}{P_{t_{in}}} \frac{V_{out}}{\sqrt{RT_{in}}} \frac{T_{in}}{T_{out}} \frac{A_{out}}{A_{in}} \quad (36)$$

where subscript “in” denote inlet to nozzle (“n”) and,

$$\left\{ \begin{array}{l} \pi_n = \frac{p_{out}}{p_{t_{in}}} \\ \tau_n = \frac{T_{out}}{T_{t_{in}}} = \pi_n^{1/a_n} \\ a_n = \frac{\gamma_n}{(\gamma_n - 1) \eta_n} \\ \eta_n = \text{nozzle polytropic efficiency} \end{array} \right. \quad (37)$$

But for adiabatic flow $T_{t_{in}} = T_{t_{out}}$, so that

$$\begin{aligned} V_{out} &= \sqrt{2C_p (T_{t_{in}} - T_{out})} = \sqrt{2C_p T_{t_{in}} (1 - \tau_n)} \\ &= \sqrt{2C_p T_{t_{in}} (1 - \pi_n^{1/a_n})} \end{aligned} \quad (38)$$

Substituting for V_{out} and τ_n into eq. 36 and simplifying gives,

$$\mu_{in} = \pi_n^{1-1/a_n} \sqrt{2a_n (1 - \pi_n^{1/a_n})} \frac{A_{out}}{A_{in}} \quad (39)$$

Neglecting $\nabla (A_{out}/A_{in})$ (i.e. for fixed nozzle geometry) and taking derivatives w.r.t.

μ_{in} ,

$$\begin{aligned} 1 &= \nabla_{\mu_{in}} \left(\pi_n^{1-1/a_n} \right) + \nabla_{\mu_{in}} \left(\sqrt{2a_n (1 - \pi_n^{1/a_n})} \right) \\ &= (1 - 1/a_n) \nabla_{\mu_{in}} \pi_n + \frac{1}{2} \nabla_{\mu_{in}} \left(2a_n (1 - \pi_n^{1/a_n}) \right) \end{aligned} \quad (40)$$

finally yields:

$$\nabla_{\mu_{in}} \pi_n = \frac{1}{1 - \frac{1}{a_n} - \frac{1}{2a_n} \frac{\pi_n^{1/a_n}}{1 - \pi_n^{1/a_n}}} = \frac{1}{1 - \frac{1}{a_n} - \frac{1}{2a_n} \frac{\tau_n}{1 - \tau_n}} \quad (41)$$

$$= \frac{2a_n(1 - \tau_n)}{(2a_n - 2) + \tau_n(1 - 2a_n)}$$

Note that π_n is the actual static to stagnation pressure ratio of the nozzle and not the applied pressure ratio. This result will be used in the following section to derive the non-restricted off-design equation. The above result holds for both the choked and unchoked condition. As we approach the critical choked conditions, $\nabla_{\mu_{in}} \pi_n$ tends to infinity and *eq. 41* gives,

$$2a_n - 2 = -\tau_n(1 - 2a_n) \Rightarrow \tau_n = \frac{2 - 2a_n}{1 - 2a_n} \quad (42)$$

If $\eta_n = 1$ (ideal nozzle) then $a_n = \gamma_n / (\gamma_n - 1)$ yielding the well known result for an ideal choked nozzle,

$$\tau_n = \frac{2}{\gamma_n + 1} \quad (43)$$

4.4.1.4 Generic Spool With Non-Restrictive Expanding Downstream Components

This section treats a generic spool where the flow downstream of the spool is not restricted. Here the outlet flow of the generic spool is expanded through a nozzle to ambient pressure and it is not possible to assume ∇_{μ_4} is negligible in *eq. 19* along the entire *ERL*, as before. *Fig. 60* compares the magnitude of various terms in *eq. 19* for the low pressure spool of the two spool unmixed turbofan which will be used as example test case for validation purposes. It is clear from the figure that in the upper region of the *ERL* (i.e. above 12 kg/s), the lp-spool is definitely restricted. However, this definite restriction does not necessarily imply that the flow at the turbine outlet is choked. In fact we know that the flow cannot be choked since the flow is choked at the exit plane of the exhaust nozzle (i.e. cannot be choked at inlet to the exhaust nozzle or there is no acceleration of the flow in the nozzle).

The portion of the *ERL* that falls within the restricted definition varies between engines and can be much smaller than illustrated for the test engine. It is therefore necessary to derive an analytical closed form solution for the unrestricted generic spool. Hence the inclusion of this section.

The stagnation pressure at inlet to nozzle is,

$$p_{t_{in}} = p_{amb} \pi_r \pi_s \quad (44)$$

Where π_r and π_s are the recovery pressure ratio prior to spool and spool pressure ratio respectively, and subscripts “in” and “amb” denote inlet to nozzle and ambient conditions respectively.

Let the static pressure at the nozzle outlet be p_{out} . The nozzle pressure ratio is then,

$$\pi_n = \frac{p_{out}}{p_{t_{in}}} = \frac{p_{out}}{p_{amb} \pi_r \pi_s} \quad (45)$$

Applying the ∇ operator to eq. 45,

$$\nabla \pi_n = \nabla_{\mu_1} \pi_n = \nabla_{\mu_1} \left(\frac{p_{out}}{p_{amb}} \right) - \nabla_{\mu_1} \pi_r - \nabla_{\mu_1} \pi_s \quad (46)$$

The first term on the RHS of eq. 46 is neglected because if the spool outlet is not restricted then it cannot be choked, in which case the nozzle flow can be fully expanded to atmospheric pressure $p_{out} = p_{amb}$ (i.e. the nozzle area can be selected for full expansion) provided the nozzle area is variable. If however, the nozzle areas are fixed, then the concerned term will not be exactly zero and it is assumed that this term is negligibly smallⁱ compared to other terms in eq. 46.

Using eq. 31 to transform the term $\nabla_{\mu_1} \pi_n$ gives,

i. During the validation phase it became clear that this assumption gives misleading results when the applied nozzle pressure ratios are very far into the choked region. See section 7.2.3 on page 198 for discussion and solution of this problem

$$\begin{aligned}
 \nabla_{\mu_1} \pi_n &= \nabla_{\mu_{in\ n}} \pi_n \times \left(1 + \nabla \left(\frac{m_{in}}{m_1} \right) + \frac{\nabla}{2} \left(\frac{T_{tin}}{T_{t_1}} \right) - \nabla \left(\frac{p_{tin}}{p_{t_1}} \right) \right) \\
 &= \nabla_{\mu_{in\ n}} \pi_n \times \left(1 + \nabla m_{rat} + \frac{\nabla}{2} \tau_s - \nabla \pi_s \right)
 \end{aligned} \tag{47}$$

Equating eq. 46 and eq. 47 and isolating $\frac{\nabla}{2} \tau_s$,

$$\begin{aligned}
 \frac{\nabla}{2} \tau_s &= \frac{-\nabla \pi_r + \nabla \pi_s \left(-1 + \nabla_{\mu_{in\ n}} \pi \right)}{\nabla_{\mu_{in\ n}} \pi} - 1 - \nabla m_{rat} \\
 &= \nabla \pi_s \left(-\frac{1}{\nabla_{\mu_{in\ n}} \pi} + 1 \right) - 1 - \nabla m_{rat} - \frac{\nabla \pi_r}{\nabla_{\mu_{in\ n}} \pi}
 \end{aligned} \tag{48}$$

Comparison of eq. 48 with eq. 19 gives the $\nabla \mu_4$ (or $\nabla_{\mu_1} \mu_4$) term which was neglected in eq. 19 for the restricted generic spool previously,

$$\nabla \mu_4 = -\frac{\nabla \pi_s}{\nabla_{\mu_{in\ n}} \pi_n} - \frac{\nabla \pi_r}{\nabla_{\mu_{in\ n}} \pi} \tag{49}$$

but,

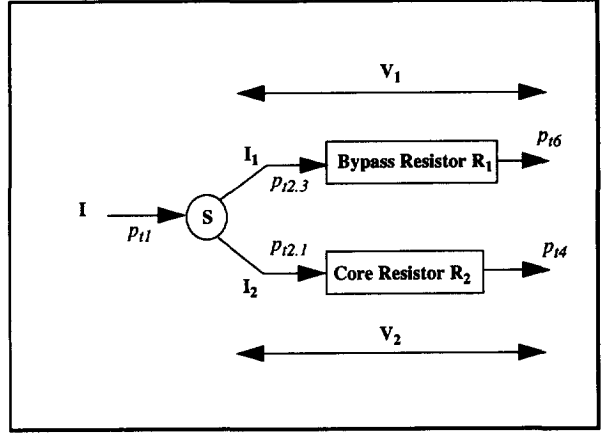
$$\pi_r = \tau_r^{a_r} \Rightarrow \nabla \pi_r = a_r \nabla \tau_r \tag{50}$$

hence,

$$\nabla \mu_4 = -\frac{\nabla \pi_s}{\nabla_{\mu_{in\ n}} \pi_n} - \frac{a_r \nabla \tau_r}{\nabla_{\mu_{in\ n}} \pi} \tag{51}$$

Using eq. 48 instead of eq. 18 and repeating the procedure exactly as for *Generic Spool with restricted Outlet* gives,

FIG. 24 Electrical Circuit Analogy of the Generic Spool for the benefit of calculating the ∇m_{rat} term, where bypass stream characteristics cannot be used or when optional mixer is used.



$$\nabla \pi_c = \frac{1 + \nabla \pi_b (-1 - C) + \frac{\nabla m_{rat}}{2} + a_r (1 - k) (\nabla \tau_r)}{B + C} \quad (52)$$

Where,

$$\begin{cases} B = 1 - \frac{\tau_c}{2a_c (\tau_c - 1)} \\ C = \frac{k - 1}{(2a_t k - 1) \left(\frac{\phi - \tau_s}{\tau_s} \right)} \\ k = 1 - \frac{1}{\nabla_{\mu_{in}} \pi_n} \end{cases} \quad (53)$$

The basic form of the equations is the same as eq. 25 but now the turbine parameters, the throttle position, and intake efficiency have an influence.

4.4.1.5 The Intake Recovery Temperature Ratio $\nabla \tau_r$ Term

The $\nabla \tau_r$ (or $\nabla_{\mu_1} \tau_r$) term in eq. 52 will have a non-zero value if there is a change in flight Mach number, e.g. when the aircraft is accelerating/decelerating or climbing/descending.

The recovery temperature is a function of flight Mach number (M_∞), namely,

$$\tau_r = 1 + \frac{\gamma - 1}{2} M_\infty^2 \quad (54)$$

Applying the ∇_{M_∞} operation,

$$\begin{aligned} \nabla_{M_\infty} \tau_r &= \frac{1}{1 + \frac{\gamma - 1}{2} M_\infty^2} \left(\cancel{1} \nabla \left(1 + \frac{\gamma - 1}{2} M_\infty^2 \right) \nabla \left(\frac{\gamma - 1}{2\gamma} M_\infty^2 \right) \right) \\ &= \frac{1}{\tau_r} (\tau_r - 1) 2 M_\infty = \frac{1}{\tau_r} (\tau_r - 1) 2 \sqrt{(\tau_r - 1) \frac{2}{\gamma - 1}} \\ &= \frac{1}{\tau_r} \left[\frac{2(\tau_r - 1)}{(\gamma - 1)^{1/3}} \right]^{3/2} \end{aligned} \quad (55)$$

But we are interested in the term $\nabla_{\mu_1} \tau_r$ (as appears in eq. 52), hence it is necessary to find $\nabla_{M_\infty} \mu_1$. The non-dimensional mass flow parameter at inlet to compressor μ_1 can be written as,

$$\begin{aligned} \mu_1 &= \frac{m \sqrt{RT_{amb}} A_\infty}{A_\infty p_{amb}} \frac{\sqrt{RT_{t_1}} p_{amb}}{A_{in} \sqrt{RT_{amb}} p_{t_1}} \\ &= \frac{m \sqrt{RT_{amb}} A_\infty}{A_\infty p_{amb}} \frac{1}{A_1 \sqrt{\tau_r \pi_r}} \end{aligned} \quad (56)$$

where m is the mass flow rate through inlet. Substituting for π_r ($\pi_r = \tau_r^{a_r}$) yields,

$$\mu_1 = \left(\frac{m \sqrt{RT_{amb}} A_\infty}{A_\infty p_{amb}} \frac{1}{A_1} \right) \tau_r^{(1/2 - a_r)} \quad (57)$$

Applying the ∇_{M_∞} operator,

$$\begin{aligned} \nabla_{M_\infty} \mu_1 &= \nabla_{M_\infty} \left(\frac{m \sqrt{RT_{amb}} A_\infty}{A_\infty p_{amb} A_1} \right) + \nabla_{M_\infty} \left(\tau_r^{(1/2 - a_r)} \right) \\ &= (1/2 - a_r) \nabla_{M_\infty} \tau_r \end{aligned} \quad (58)$$

Where it is assumed that $\nabla_{M_\infty} (\rho_1 V_1)$ is negligibly small compared to other terms in eq. 58. Substituting for $\nabla_{M_\infty} \tau_r$ from eq. 55 into eq. 58 gives,

$$\begin{aligned} \nabla_{M_\infty} \mu_1 &= (1/2 - a_r) \frac{1}{\tau_r} \left[\frac{2(\tau_r - 1)}{(\gamma - 1)^{1/3}} \right]^{3/2} \\ &= (1/2 - a_r) \frac{1}{\tau_r} \left[\frac{2(\tau_r - 1)}{v(\gamma - 1)^{1/3}} \right]^{3/2} \end{aligned} \quad (59)$$

The above gives the change in non-dimensional mass flow parameter at compressor inlet f for any change in flight mach number. Rearranging eq. 58 gives,

$$\nabla_{\mu_1} \tau_r = \frac{\nabla_{M_\infty} \tau_r}{\nabla_{M_\infty} \mu_1} = \frac{1}{(1/2 - a_r)} \quad (60)$$

4.4.1.6 Generic Spool with Well Designed and Well Matched Components

The relations developed so far are dependent on how the flow is restricted downstream of the *Generic Spool*. In practice the inner spools are adequately treated by eq. 25 (restricted outlet) and the outermost spool by eq. 52 (outlet coupled to expanding nozzle). However, if minimal information about turbine and compressor stages is available, (i.e. type of stage and number of stages) then off-design equations can be derived regardless of how the flow is restricted downstream of the spool.

The flow coefficient at entry to a turbo-machine is, $\phi = \frac{V_Z}{U}$ where V_Z is the average axial flow velocity and U is the tangential wheel speed at the mean radius. It can simply be shown that,

$$\phi = \frac{\frac{m\sqrt{T_t}}{Ap_t} \frac{\rho_t}{\rho}}{\frac{U}{R\sqrt{T_t}}} \quad (61)$$

The ratio of ϕ 's of turbine to compressor is,

$$\frac{\phi_t}{\phi_c} = \left(\frac{\frac{m\sqrt{T_t}}{Ap_t} \frac{\rho_t}{\rho}}{\frac{U}{R\sqrt{T_t}}} \right)_3 / \left(\frac{\frac{m\sqrt{T_t}}{Ap_t} \frac{\rho_t}{\rho}}{\frac{U}{R\sqrt{T_t}}} \right)_1 = \frac{\mu_3}{\mu_1} \sqrt{\phi} \left(\frac{\rho_t}{\rho} \right)_3 / \left(\frac{\rho_t}{\rho} \right)_1 = \frac{\mu_3}{\mu_1} \sqrt{\phi} \quad (62)$$

$$\therefore \phi = \left(\frac{\mu_1 \phi_t}{\mu_3 \phi_c} \right)^2 \quad (63)$$

Where $\frac{\rho_{t_3}}{\rho_{t_1}}$ has been assumedⁱ to equal $\frac{\rho_3}{\rho_1}$.

In practice it is possible to design compressor and turbines which are well designed and well matched. It appears from the *Equilibrium Running Line (ERL)* of most spools that the designers have chosen the right combination of parameters so that the *ERL* closely follows the *Back Bone*ⁱⁱ of the compressor and/or is almost parallel to it, for a significant portion of the *ERL*. For fans, however, this may require variable geometry, e.g. variable pitch fan and/or variable nozzle area.

i. it is not really necessary to assume $\rho_{t_3}/\rho_{t_1} = \rho_3/\rho_1$ because eq. 63 will be differentiated at a later stage in which case it would be necessary to assume that $\nabla \left(\rho_{t_3}/\rho_{t_1} + \rho_3/\rho_1 \right)$ is negligibly small compared to other terms in eq. 70. This practically the same as assuming $\nabla_{\mu_1} \mu_3$ is negligibly small.

ii. *Back Bone of the compressor is the maximum efficiency line on the compressor map.*

An estimate of the backbone of the compressor is obtained by assuming that the flow coefficient of the first stage of compressor is constant at its design point value, i.e. constant ϕ_c in eq. 63.

The ϕ_t term is the flow coefficient of the first turbine stage. The variation of this term with the stage work done factor depends on the type of turbine stage (i.e. velocity triangles) as shown below,

$$\phi_t = E\Psi_{t_s} + F \quad (64)$$

Where Ψ_{t_s} is the turbine stage work done factor, E and F are constants obtained from the velocity triangles^[23], i.e. type of turbine stage,

$$\begin{aligned} \frac{F}{E} &= 2 \quad \text{For Impulse Stage} \\ \frac{F}{E} &= 1 \quad \text{For Symetric Stage} \\ \frac{F}{E} &= 0 \quad \text{For Zero Swirl Stage} \end{aligned} \quad (65)$$

If the mean radius of the turbine is constant and equal work is done in N_t turbine stages then,

$$\Psi_{t_s} = \frac{\Psi_t}{N_t} = \frac{C_{p3} T_{t3} (1 - \tau_t)}{U^2 N_t} \quad (66)$$

Substituting for ϕ_t in eq. 63,

$$\phi = \left(\frac{\mu_1}{\mu_3} \frac{E\Psi_{t_s} + F}{\phi_c} \right)^2 \quad (67)$$

But Ψ_{t_s} is related to the compressor work done factorⁱ Ψ_c i.e.,

$$\frac{\Psi_t}{\Psi_c} = \frac{\frac{C_{p3} T_{t3} (1 - \tau_t)}{U^2}}{\frac{C_{p1} T_{t1} (\tau_c - 1)}{U^2}} \quad (68)$$

Using eq. 66 and eq. 5,

$$\begin{aligned} \frac{\Psi_t}{\Psi_c} &= \frac{N_t \Psi_{t_s}}{\Psi_c} = C_{p_{rat}} \frac{(\tau_c - 1) / \left(C_{p_{rat}} m_{rat} \emptyset \right)}{\tau_c - 1} = \frac{1}{m_{rat}} \\ &\Rightarrow \Psi_{t_s} = \frac{\Psi_c}{N_t m_{rat}} = \frac{N_c \Psi_{c_s}}{N_t m_{rat}} \end{aligned} \quad (69)$$

Ψ_c can be replaced by $N_c \Psi_{c_s}$ if repeating compressor stagesⁱ are assumed. Applying the ∇ operator to eq. 67,

$$\nabla \emptyset = 2 \left(1 - \nabla \mu_3 - \nabla (E \Psi_{t_s} + F) \right) \quad (70)$$

Where $\nabla \mu_3$ is neglected because it is small compared to other terms in eq. 70. Substituting for Ψ_{t_s} from eq. 69 into eq. 70 and simplifying,

-
- i. in principle there should be a $(d_2/d_1)^2$ multiplier on the RHS of eq. 68 but it disappears after the differentiation of this equation
 - i. repeating stages are stages with equal work done factor

$$\nabla\emptyset = 2 \left(1 - \frac{\nabla m_{rat}}{1 + \frac{F/E}{\Psi_{t_s}}} \right) = 2 \left(1 - \frac{\nabla m_{rat}}{1 + \frac{F/E}{\left(\frac{N_c \Psi_{c_s}}{N_t m_{rat}} \right)}} \right) \quad (71)$$

Here minimal information about the type and number of stages has lead to a $\nabla\emptyset$ independent of $\nabla\pi_c$ and $\nabla\pi_b$ (c.f. eq. 12). All terms on the RHS are constant except ∇m_{rat} and m_{rat} . Designating G ,

$$G = \frac{1}{1 + \frac{F/E}{\left(\frac{N_c \Psi_{c_s}}{N_t m_{rat}} \right)}} \quad (72)$$

yields,

$$\nabla\emptyset = 2 (1 - G \nabla m_{rat}) \quad (73)$$

Where $G \geq 1$. Equating this to $\nabla\emptyset$ from eq. 12,

$$\nabla\emptyset = 2 (1 - G \nabla m_{rat}) = 2 (-1 + \nabla\pi_c + \nabla\pi_b - \nabla m_{rat} + \nabla\mu_3) \quad (74)$$

Rearranging and isolating $\nabla\pi_c$,

$$\nabla\pi_c = 2 + \nabla m_{rat} (1 - G) - \nabla\pi_b \quad (75)$$

Which is a simple relation independent of the downstream component, for a “Well Designed and Well Matched Spool”.

The $\nabla\pi_b$ term is w.r.t. μ_1 . Using transformation eq. 30 in order to obtain $\nabla_{\mu_{2.1}} \pi_b$ and substituting into eq. 75,

$$\nabla \pi_c = 2 + \nabla m_{rat} (1 - G) - 2 \left(\nabla_{\mu_{2.1}} \pi_b \right) \left(1 + \nabla m_{rat} + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right) \quad (76)$$

Isolating $\nabla \pi_c$,

$$\nabla \pi_c = \frac{2 - \nabla_{\mu_{2.1}} \pi_b + \nabla m_{rat} \left(1 - G - \nabla_{\mu_{2.1}} \pi_b \right)}{1 + \nabla_{\mu_{2.1}} \pi_b \left(\frac{1}{2a_c} - 1 \right)} \quad (77)$$

The above result is valid regardless of what the downstream flow conditions. The absence of turbine efficiency and throttle position is also noted here.

4.4.1.7 The Mass Ratio Term from Bypass Nozzle Characteristics

The ∇m_{rat} term, appearing in all equations becomes very significant particularly for the fan spool where the engine bypass ratio varies significantly along the working line of the engine, or for spools where the variation in cooling flow along the *ERL* is significant.

If the spool is the outermost *LP* spool of an unmixed turbofan, then the ∇m_{rat} can be found non-iteratively by considering the flow characteristics of the bypass nozzle.

From eq. 39,

$$\nabla_{\pi_n} \mu_{in} = \frac{1}{\nabla_{\mu_{in}} \pi_n} = \frac{(2a_n - 2) + \tau_n (1 - 2a_n)}{2a_n (1 - \tau_n)} \quad (78)$$

Where subscript “*in*” refers to the inlet of bypass nozzle.

Also the gradient of the bypass nozzle pressure ratio,

$$\pi_n = \frac{P_{exit}}{P_{atm} \pi_r \pi_c} \Rightarrow \nabla \pi_n = \nabla \left(\frac{P_{exit}}{P_{atm}} \right) - (\nabla \pi_r + \nabla \pi_c) \quad (79)$$

neglected

Now,

$$\begin{aligned}\nabla \mu_{in} &= \nabla_{\mu_1} \mu_{in} = \nabla_{\pi_n} \mu_{in} \times \nabla_{\mu_1} \pi_n = -\nabla_{\pi_n} \mu_{in} (\nabla \pi_r + \nabla \pi_c) \\ &= 1 + \nabla \left(\frac{m_{byp}}{m_1} \right) + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right)\end{aligned}\quad (80)$$

Where the right most side was taken from eq. 31. Isolating the mass term,

$$\nabla \left(\frac{m_{byp}}{m_1} \right) = -\nabla_{\pi_n} \mu_{in} (\nabla \pi_r + \nabla \pi_c) - 1 - \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \quad (81)$$

But,

$$m_{rat} = \frac{m_3}{m_1} = 1 - \frac{m_{byp}}{m_1} \quad (82)$$

Applying the ∇ operation,

$$\begin{aligned}\nabla m_{rat} &= -\frac{m_{byp}/m_1}{1 - m_{byp}/m_1} \nabla \left(\frac{m_{byp}}{m_1} \right) \\ &= \frac{1 - m_{rat}}{m_{rat}} \left(\nabla_{\pi_n} \mu_{in} (\nabla \pi_r + \nabla \pi_c) + 1 + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right)\end{aligned}\quad (83)$$

or replacing the $\nabla_{\pi_n} \mu_{in}$ term from eq. 78,

$$\nabla m_{rat} = \frac{1 - m_{rat}}{m_{rat}} \left(\frac{(2a_n - 2) + \tau_n (1 - 2a_n)}{2a_n (1 - \tau_n)} (\nabla \pi_r + \nabla \pi_c) + 1 + \nabla \pi_c \left(\frac{1}{2a_c} - 1 \right) \right) \quad (84)$$

Note that the term $(1 - m_{rat})/m_{rat}$ is simply the bypass ratio λ .

4.4.1.8 The Mass Ratio Term from Electrical Analogy

If it is not possible to use bypass nozzle characteristics (i.e not *LP* spool of an unmixed turbofan), then the mass ratio term can still be found by using analogy with electrical circuits. The Analogy is made between 1-D mass flow characteristics *eq. 26* of a duct and the famous $V=RI$ relation, i.e.,

$$1 - \pi = \frac{\Delta p_t}{q} \frac{1}{2} \mu_{in}^2 = \left(\frac{\Delta p_t}{q} \frac{1}{2} \right) \left(\frac{m \sqrt{RT_t}}{AP_t} \right)_{in}^2 \quad (85)$$

which corresponds to the electrical equation

$$V = R \times I \quad (86)$$

Where subscript “in” refers to duct inlet. If the square of the mass flux parameter μ^2 at generic spool stations 1, 2.1 and 2.3 are considered as electrical currents I , I_2 , I_1 respectively then the electrical analogy yields,

$$\begin{aligned} \nabla m_{rat} &= \nabla (\mu_{2.1}^2 / \mu_1^2)^{0.5} = \nabla \left(\frac{I_2}{(I_2 + I_1)} \right)^{0.5} = \nabla \left(\frac{1}{1 + \frac{I_1}{I_2}} \right)^{0.5} \\ &= -\frac{1}{2} \frac{I_1/I_2}{1 + I_1/I_2} (\nabla I_1 - \nabla I_2) \end{aligned} \quad (87)$$

where the voltages corresponding to the I_1 and I_2 are,

$$\begin{aligned} V_1 &= 1 - \frac{p_{t_6}}{p_{t_{2.3}}} = 1 - \frac{1}{\pi_c} \frac{p_{t_6}}{p_{t_1}} \\ V_2 &= 1 - \frac{p_{t_4}}{p_{t_{2.1}}} = 1 - \frac{1}{\pi_c} \frac{p_{t_4}}{p_{t_1}} \end{aligned} \quad (88)$$

Now for constant resistance R ,

$$\nabla I = \nabla \left(\frac{V}{R} \right) = \nabla V - \nabla R = \nabla V \quad (89)$$

and,

$$\nabla V = \nabla (1 - \pi) = -\frac{\pi}{1 - \pi} \nabla \pi \quad (90)$$

where,

$$\pi_1 = \frac{1}{\pi_c} \frac{P_{t_6}}{P_{t_1}}, \pi_2 = \frac{1}{\pi_c} \frac{P_{t_4}}{P_{t_1}} \quad (91)$$

and,

$$\nabla \pi_1 = -\nabla \pi_c + \frac{\nabla P_{t_6}}{P_{t_1}}, \nabla \pi_2 = -\nabla \pi_c + \frac{\nabla P_{t_4}}{P_{t_1}} \quad (92)$$

Here the terms $\nabla \left(P_{t_6} / P_{t_1} \right)$ and $\nabla \left(P_{t_4} / P_{t_1} \right)$ have been assumed to be negligible compared to the $\nabla \pi_c$ term.

Combining eq. 89 and eq. 90 and substituting for two values of π gives,

$$\begin{aligned} \nabla I_1 &= -\frac{1}{\pi_c} \frac{P_{t_6}}{P_{t_1}} \left/ \left(1 - \frac{1}{\pi_c} \frac{P_{t_6}}{P_{t_1}} \right) \right. \times -\nabla \pi_c \\ \nabla I_2 &= -\left(\frac{1}{\pi_c} \frac{P_{t_4}}{P_{t_1}} \right) \left/ \left(1 - \frac{1}{\pi_c} \frac{P_{t_4}}{P_{t_1}} \right) \right. \times -\nabla \pi_c \end{aligned} \quad (93)$$

Using the electrical identity, $I = I_1 + I_2$ then

$$\frac{I_1 / I_2}{1 + I_1 / I_2} = \frac{I_1}{I_2 + I_1} = \frac{I - I_2}{I} = 1 - \frac{I_2}{I} = 1 - m_{rat}^2 \quad (94)$$

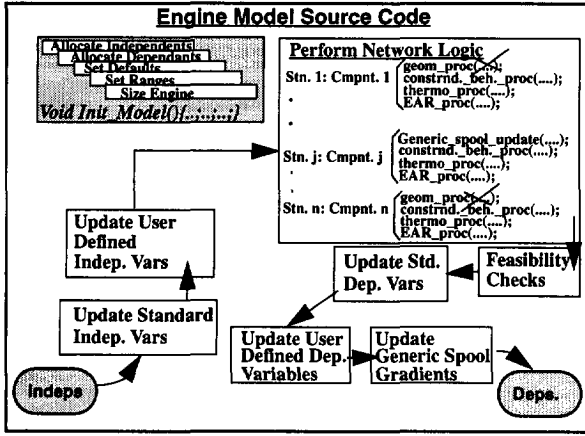


FIG. 25 Structure of Off-Design Engine Model generated by CAGED. It is very similar to the On-Design Model (Fig. 19) except that a Constrained Behaviour Process (Fig. 15) is called before each thermodynamic process and modules are added for calculation and implementation of direct off-design solution for spools. The geometry process at each component can be eliminated if the type of component and its size is to remain fixed.

Finally, substituting for $(I_1/I_2)/(1 + I_1/I_2)$, ∇I_1 and ∇I_2 from eq. 94 and eq. 93 respectively into eq. 87 yields,

$$\nabla m_{rat} = -\frac{1}{2} (1 - m_{rat}^2) \left(\left(\frac{1}{\pi_c} \frac{p_{t6}}{p_{t1}} \right) / \left(1 - \frac{1}{\pi_c} \frac{p_{t6}}{p_{t1}} \right) - \left(\frac{1}{\pi_c} \frac{p_{t4}}{p_{t1}} \right) / \left(1 - \frac{1}{\pi_c} \frac{p_{t4}}{p_{t1}} \right) \right) \nabla \pi_c \quad (95)$$

or,

$$\nabla m_{rat} = -\frac{1}{2} (1 - m_{rat}^2) \left(\frac{P_{t6}/P_{t2}}{1 - P_{t6}/P_{t2}} - \frac{P_{t4}/P_{t2}}{1 - P_{t4}/P_{t2}} \right) \nabla \pi_c \quad (96)$$

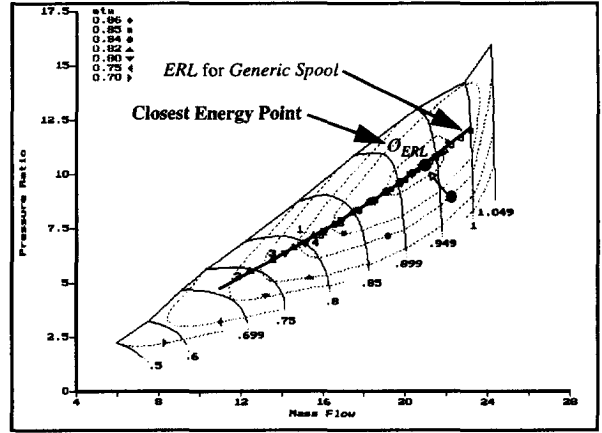
4.4.1.9 Correction for Power Off-Take and Variation of Compressor Efficiency

The efficiency of the compressor varies with μ_1 and has significant effect on the major gradients given above. If η_c (or a_c) varies then the polytropic relation $\pi_c = \tau_c^{a_c}$ yields,

$$\nabla \pi_c = a_c \nabla \tau_c + \tau_c \nabla a_c \log(\tau_c) = a_c \nabla \tau_c + \tau_c \nabla a_c \log(\pi_c) \quad (97)$$

instead of $\nabla \pi_c = a_c \nabla \tau_c$ used in all the derivations above.

FIG. 26 Determination of shaft torque from the Equilibrium Running Line of Generic Spool, for transient performance calculations. Energy difference from the current spool operating point O_I to the closest (energy wise) steady state point O_{ERL} gives the current shaft torque. (Plot from *GASTURB*). The x-axis represents corrected mass flow parameter $(m\sqrt{\theta})/\delta$ (kg/s).



In addition, if power is taken from the spool, then the power balance relation *eq. 4* does not hold any longer and has to be modified. However modification of the above derivation leads to complicatedⁱ equations and the author prefers to apply a two step correction to the $\nabla\pi_c$ values that are calculated for constant polytropic efficiency and zero power off-take,

$$(\nabla\pi_c)_{intermediate} = \nabla\pi_c - \nabla a_c \log(\pi_c) \quad (98)$$

$$(\nabla\pi_c)_{corrected} = (\nabla\pi_c)_{intermediate} \left(\frac{\tau_c}{(\tau_c + P_{OffTake} / (m_1 C_{p1} T_{t1}))} \right)^{a_c} \quad (99)$$

The intermediate correction above for variation of compressor efficiency works for cases where the variation is relatively small.

4.4.1.10 Inclusion of a Generic Spool within Another Generic Spool

Fig. 23 shows that the burner component of the generic spool (*Fig. 22*) can be replaced by another generic spool and this can be recursively continued in order to represent

- i. It became clear during the validation phase that efficiency variation along the ERL can be relatively large (e.g. for a low pressure spool at high and low values of fan pressure ratio) so that the simple correction was not sufficiently accurate. Hence the derivation of the closed form equation were repeated but this time using *eq. 98* to relate compressor pressure and temperature ratio gradients. See section 7.2 on page 174 for further discussion.

multiple spool engines. The term that couples the inner spool to the outer spool is the $\nabla\pi_b$ term which appears in the previously derived sensitivity equations (e.g. eq. 75). For multiple spools, the calculation starts from the innermost spool outwards. The innermost spool will then include a real burner and the $\nabla\pi_b$ term is calculated using eq. 32. Once the innermost spool is calculated, then enough information is available to proceed to the enclosing spool. Here the $\nabla\pi_b$ term comes from the sensitivity of the inner spool pressure ratio $(\nabla\pi_{s_{InnerSpool}})^i$, i.e.

$$\left(\nabla_{\mu_1} \pi_b\right)_{Enclosing Spool} \equiv \left(\nabla_{\mu_1} \pi_s\right)_{InnerSpool} \quad (100)$$

Where the RHS comes from eq. 17 wherein inner spool parameters are substituted.

4.4.2 Summary of Direct Analytical Off-Design Equations in Closed Form

We now have a sufficient collection of relations to state the off-design equations in *Closed Form*. In their original form, the above equations are useful for seeing the influence of the various terms, $(\nabla m_{rat}, \nabla\pi_b, \nabla a_c \dots \text{etc.})$ on $\nabla\pi_c$. In their original form however it is not apparent that the equations are **Closed** and hence suitable for computer programming. For example, eq. 25 gives $\nabla\pi_c$ for generic spool with restrictive outlet, in terms of ∇m_{rat} . A glance at eq. 96 shows that ∇m_{rat} term is itself in terms of $\nabla\pi_c$, appearing therefore *Unclosed* or *Circular*. In this section all the above set of relations will be collected and tabulated, to prove that the solution is in closed form and make the equations fit for direct implementation in a computer program.

In general the $\nabla\pi_c$ equations for the *Generic Spool* can be written as follows,

$$\nabla\pi_c = \frac{Z + \sum_{i=\pi_r}^{m_{rat}} (X_i \nabla\pi_c + Y_i)}{B + C} \quad (101)$$

i. Care must be taken with mixing only the gradients with respect to the same μ , Transformation eq. 30 or eq. 31 should be used.

Where i is the name of the variable with influence (i.e. $i = \pi_r, \pi_b \dots etc$), B and C keep their original definitions, X_i and Y_i are then influence coefficients which will be tabulated with the aid of the above derivations. Extracting $\nabla \pi_c$ then,

$$\nabla \pi_c = \frac{Z + \nabla \pi_c \sum_{i=\pi_r}^{m_{rat}} X_i + \sum_{i=\pi_r}^{m_{rat}} Y_i}{B + C} \Rightarrow \nabla \pi_c = \frac{Z + \sum_{i=\pi_r}^{m_{rat}} Y_i}{B + C - \sum_{i=\pi_r}^{m_{rat}} X_i} \quad (102)$$

The above closed form solution is summarized in *Table 6* (in appendix) showing the contribution of major spool parameters to the overall solution. The table gives a two step correction for variation in efficiency and shaft power off-take (explained in section 4.4.1.9 on page 83) so that they are valid for relatively small fan/compressor efficiency variation along the *ERL* (i.e. relatively flat compressor efficiency curves). A more rigorous set of solutions is possible by incorporating the ∇a_c term in the derivations from the start, as given in section 7.2.1 on page 177.

Several important points should be noted at this stage,

1. **Non-Dimensionality:** The non dimensional gradients are non-dimensionalized *with respect to the current operating point of the Generic Spool* and not the design point or some fixed reference point.
2. **Closed Form:** With the aid of the *Closed Form* solutions above, the gradients $\nabla \pi_c$, $\nabla \theta$ and $\nabla \pi_s$ can be calculated in terms of generic spool parameters at the *current* steady state operating point. As long as the generic spool remains in its steady state, any steady state departure from a current operating are governed by the closed form equations given above.
3. **Restricted Outlet Issue:** The outlet flow of a generic spool is restricted, if the flow has to still pass through several draggy components before finally being released into the atmosphere. To be precise, spool outlet is restricted if $\nabla \mu_4$ term is very small compared to other terms in *eq. 18*. It is not necessary for flow to be choked at station 4 for this condition to hold. $Mach_4$ could be say 0.8, and $\nabla \mu_4$ could still be negligible if resistance (i.e. drag) to flow is high between station 4 and engine exhaust. Internal Spools, e.g. *IP* and *HP* spools certainly

satisfy this “restricted outlet” condition, because there are still many draggy downstream components and increasing pressure at station 4 gives relatively small change in μ_4 (see equation eq. 26 for link between pressure ratio and mass flow parameter for a general duct). The outlet of *LP* spools are not always restricted, because the flow only has to pass through an exhaust nozzle before being released into atmosphere. In this case $\nabla\mu_4$ term in eq. 18 is not always negligible and is determined by the flow characteristics of the nozzle. If minimal information is available about the type of first turbine stage then, the *Well Designed Well Matched* assumption gives solution *regardless* of outlet restriction.

4. The $\nabla\pi_s$ calculated for the generic spool serves as $\nabla_{\mu_2}\pi_b$ for the enclosing spool. This allows recursive inclusion of spools (Fig. 23).

4.4.3 Strategy for using the *Non-Iterative Off-Design Equations* and Source Code Generation for Off Design

CAGED generates source code for high speed non-iterative calculation of the *ERL* of user defined engine configurations automatically, so that the *HD* does not have to bother with this aspect. This section is however included to show how *CAGED* does this.

The basic structure of the *Off-Design Engine Model* generated by *CAGED* is visualized in Fig. 25 . The structure is basically the same as the *On-Design Engine Model* (Fig. 19) with the following exceptions,

1. The function *Void init_model(.....)*; now includes the sizing of the engine which is done only once. Here the mass flow rate through the engine is fixed. If the engine flow path areas are to remain fixed (i.e. not variable geometry engine model), then all flow path areas are calculated here, from the available on-design fluid properties and selected mass flow rate. Else, the areas that are to be fixed are calculated here, and the variable areas are recalculated in function *Void gt(.....)*; at entry to each component.
2. In performing *Network Logic*, the constrained behaviour process (See section 4.1.1) is called to determine the components thermodynamic variables, e.g. this could be reading component maps. If a components geometry is variable, then a geometry process is called before every constrained behaviour process call, else it is omitted.
3. If a component *j* is the compressor of a *Generic Spool* then generic spool parameters are updated (i.e. π and ϕ) using the new incoming flow properties, and previously calculated gradients, i.e.

FIG. 27 a) Turbojet mission analysis. Design point at SLS and $TET=1200^{\circ}K$. All point other than design point are calculated by the *Off-design* engine model. (Curves Generated by GASTURB)

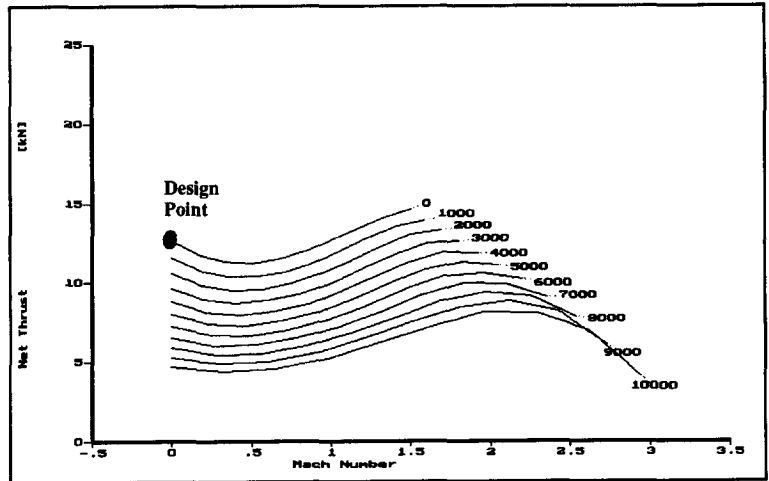


FIG. 27 b) Effect of changing the design point value of TET to $1400^{\circ}K$

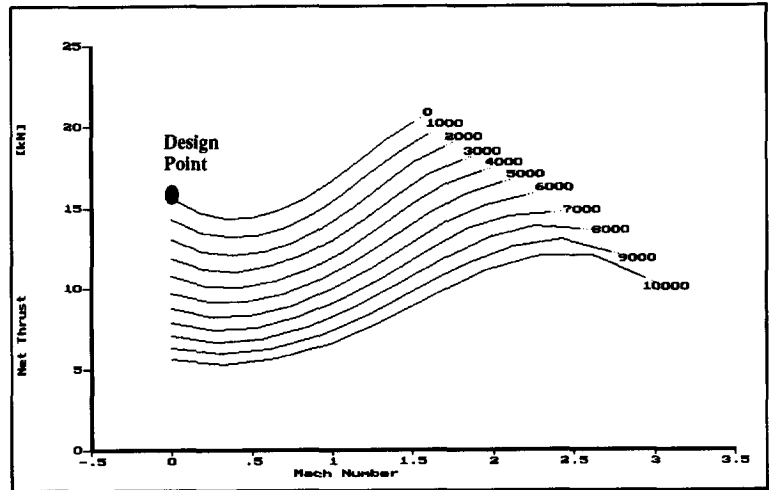
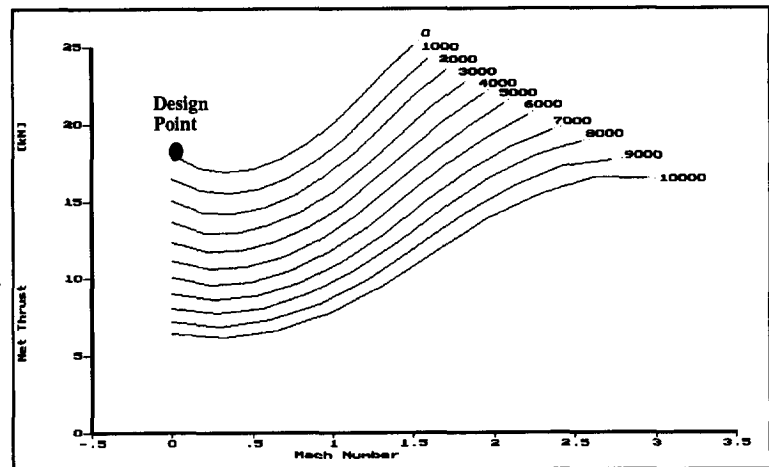


FIG. 27 c) Effect of further increasing TET to $1600^{\circ}K$. Since *CAGED* Off-Design Engine models are high speed, the human observes the information update between figures a) to b) to c) *instantaneously*.



$$\begin{aligned}
 \pi_{c_{new}} &= \pi_{c_{old}} + \left(\mu_{1_{new}} - \mu_{1_{old}} \right) \frac{\pi_{c_{old}}}{\mu_{1_{old}}} \nabla \pi_{c_{old}} \\
 &\text{and} \\
 \emptyset_{new} &= \emptyset_{old} + \left(\mu_{1_{new}} - \mu_{1_{old}} \right) \frac{\emptyset_{old}}{\mu_{1_{old}}} \nabla \emptyset_{old}
 \end{aligned}
 \tag{103}$$

4. After dependent variables have been updated, then generic spool gradients ($\nabla \pi_c, \nabla \emptyset, \nabla \pi_s$) are updated for the next run of function *Void gt(...,...)*. If spools are recursively included, (e.g. *Fig. 23*), then updating starts from the innermost spool outwards. Here the $\nabla \pi_s$ value of an inner spool becomes the $\nabla_{\mu_2} \pi_b$ for the enclosing spool. Then the enclosing spool gradients are calculated and so on.
5. The choice of which column of Table 3 to use for calculation of spool gradients, is as follows,
 - If spool is an inner spool, e.g. *HP* or *IP* spools, where flow has to still pass through several downstream components before being released into atmosphere, then can use the restricted outlet form, the first column.
 - If spool is an outermost spool, e.g. *LP* spool, then use the unrestricted form. If bypass flow is simple, i.e. simple bypass nozzle releasing into atmosphere, then use m_{rat} term from bypass nozzle. Else if bypass flow is not a simple nozzle, then use the electrical analogy form for m_{rat} term.
 - If the type of compressor and turbine stage are known, then one can use the *Well Designed Well Matched Spool* regardless of the type of restriction at outlet of the spool.

In *step 3* above, the next point on the *ERL* is found by extrapolation from the current point using the calculated gradient of the *ERL* curve there. As the *ERL* is not exactly a straight line, then the next point will not fall exactly on the *ERL*. Since the whole of the *ERL* is calculated in this manner, we would expect the errors to accumulate as the calculation proceeds down along the *ERL*.

A probably better alternative to *step 3* would be to integrate the calculated sensitivity $\nabla \pi_{c_{old}}$ as shown below,

$$\nabla \pi_{c_{old}} = \frac{\partial \pi_c / \pi_c}{\partial \mu_1 / \mu_1} \Rightarrow \int_{\pi_{c_{old}}}^{\pi_{c_{new}}} \partial \pi_c / \pi_c = \nabla \pi_{c_{old}} \int_{\mu_{1_{old}}}^{\mu_{1_{new}}} \partial \mu_1 / \mu_1 \quad (104)$$

which yields the alternative for *step 3* above,

$$\boxed{\begin{aligned} \pi_{c_{new}} &= \pi_{c_{old}} \left(\frac{\mu_{1_{new}}}{\mu_{1_{old}}} \right)^{\nabla \pi_{c_{old}}} \\ \text{and similarly} \\ \varnothing_{new} &= \varnothing_{old} \left(\frac{\mu_{1_{new}}}{\mu_{1_{old}}} \right)^{\nabla \varnothing_{old}} \end{aligned}} \quad (105)$$

So that now there is a non-linear extrapolation for each segment of the *ERL* instead of linear extrapolation given in *step 3* above.

Now the above equation is also particularly interesting for deriving a back of the envelope (i.e. not so accurate) for calculation of the whole of the *ERL* in one go and not in segments as shown below,

$$\begin{aligned} \pi_{c_{new}} &= \pi_{c_{dp}} \left(\frac{\mu_{1_{new}}}{\mu_{1_{dp}}} \right)^{\nabla \pi_{c_{dp}}} \\ \text{and similarly} \\ \varnothing_{new} &= \varnothing_{dp} \left(\frac{\mu_{1_{new}}}{\mu_{1_{dp}}} \right)^{\nabla \varnothing_{dp}} \end{aligned} \quad (106)$$

Where the subscript “*dp*” refers to the design point conditions where all spool parameters are known and the sensitivities can be calculated with the aid of the closed form solutions given in this report. In this way the whole of the *ERL* can be roughly calculated *using design point information only*.

4.5 Transient Behaviour of a Generic Spool

The transient behaviour of an engine can be a conceptual/preliminary design requirement. For example the time to accelerate can be a critical factor in selecting the best engine design. Hence the inclusion of this section.

Quasi-steady transient analysis (like the iterative steady state off-design analysis) is explained very adequately in standard propulsion texts and implemented with high accuracy in various thermodynamic analysis packages. However, the solution procedure is usually iterative, with each step involving interrogation of compressor, turbine and other components maps. Reading the compressor and the turbine maps is necessary because, *the torque responsible for acceleration/deceleration of spool comes from the difference in power between the current compressor operating point and the current turbine operating point on their respective maps*. The iterative nature of solution and the necessity to interrogate component maps implies that full transient performance calculations can be high speed only by use of special hardware and/or computers^[59].

However, it is possible to derive fast transient models by looking at usually ignored information which is contained in the steady state *ERL* of the engine. This chapter has shown how the *ERL* of a generic spool can be calculated at high speed at any given engine operating conditions and engine settings. The piece of information usually ignored in transient analysis is that the relative position of the current operating point of the spool and the *ERL* gives enough information to calculate the shaft torque, leading to a high speed non-iterative alternative for transient analysis of spools.

Consider a generic spool at a transient operating point O in the compressor map as shown in Fig. 26. Since O is a transient point, it is characterized by values π_c , ω , and μ_1 as well as their time rates. In other words, at a given point in time, the *ERL* is given (i.e. $\pi_c, a_c, U_c, \emptyset$ is known as function of mass flow parameter at compressor inlet μ_1) and the following time rates are known,

$$\frac{\partial \omega}{\partial t}, \frac{\partial \pi_c}{\partial t}, \frac{\partial \mu_1}{\partial t} \text{ and ERL given} \quad (107)$$

The goal of transient analysis is to find the new position of O (i.e. new π_c , ω , and μ_1) and new values of the time rates $\dot{\omega}$, $\dot{\pi}_c$ and $\dot{\mu}_1$ after an infinitesimal time lapse of δt .

All the points on the ERL represent the operating points of the spool where the compressor and the turbine are matched. The ERL therefore tries to attract O towards itself. The strongest attraction comes from the point O_{ERL} which has the closest energy state to the point O . O_{ERL} is the point whose sum of rotational and fluid energy rates are closest to that of point O . Stated mathematically, O_{ERL} is the point on the ERL whose \dot{e} is closest to that of current transient point O , where

$$\begin{aligned}\dot{e} &= \frac{\partial}{\partial t} \left(I \frac{\omega^2}{2} + m c_p T_1 (\tau_c - 1) \right) \\ &= I \omega \dot{\omega} + \dot{m} c_p T_1 (\tau_c - 1)\end{aligned}\quad (108)$$

The difference $\Delta \dot{e} = \dot{e}_{O_{ERL}} - \dot{e}_O$ gives the torque increment $\Delta \Upsilon$ and hence the increment in angular acceleration $\Delta \dot{\omega}$ i.e.

$$\Upsilon = \frac{\text{Shaft Power}}{\omega} \Rightarrow \Delta \Upsilon = \frac{\Delta \dot{e}}{\omega} = \frac{\dot{e}_{O_{ERL}} - \dot{e}_O}{\omega} \quad (109)$$

and since $\Upsilon = I \dot{\omega}$

$$\Delta \dot{\omega} = \frac{\Delta \Upsilon}{I} = \frac{\dot{e}_{O_{ERL}} - \dot{e}_O}{I \omega} \quad (110)$$

Now the time taken Δt for O to reach O_{ERL} , all things being the same is given by,

$$\Delta t = \frac{\omega_{ERL} - \omega}{\Delta \dot{\omega}} \quad (111)$$

Which can be used to update the time rates $\dot{\omega}$, $\dot{\pi}_c$ and $\dot{\mu}_1$ i.e.

$$\begin{aligned}
 \dot{\omega}_{new} &= \dot{\omega}_{old} + \frac{\omega_{O_{ERL}} - \omega_O}{\Delta t} \\
 \dot{\pi}_{c_{new}} &= \dot{\pi}_{c_{old}} + \frac{\pi_{c_{O_{ERL}}} - \pi_{c_O}}{\Delta t} \\
 \dot{\mu}_{1_{new}} &= \dot{\mu}_{1_{old}} + \frac{\mu_{O_{ERL}} - \mu_O}{\Delta t}
 \end{aligned}
 \tag{112}$$

The new position of O (i.e. new values of π_c , ω and μ_1) are then calculated from the above new rates as follows,

$$\begin{aligned}
 \pi_{c_{new}} &= \pi_{c_{old}} + \dot{\pi}_{c_{new}} \delta t \\
 \omega_{new} &= \omega_{old} + \dot{\omega}_{new} \delta t \\
 \mu_{1_{new}} &= \mu_{1_{old}} + \dot{\mu}_{1_{new}} \delta t
 \end{aligned}
 \tag{113}$$

Therefore only the following pieces of information are needed to enable transient analysis.

1. definition of *ERL* of the spool
2. definition of current operating point of spool O
3. spool moment of inertia
4. choice of infinitesimal time lapse δt

The choice of δt is important. It must be small enough (compared with characteristic acceleration time) for the above trends to hold, and large enough so that calculation effort will not be too high.

In general the *ERL* of an engine is not fixed but varies as engine parameters change, for example,

1. as operating conditions vary (e.g. flight Mach number and altitude)
2. as some components go through geometrical change (e.g. variable exhaust nozzle)
3. as amount of bleed varies
4. as power off-take from spool varies

The direct non-iterative off design relations developed in previous sections, then provide the means for recalculating a new *ERL* every time this is necessary. Also the definition of point *O* can go through sudden changes. These variation do not provide difficulties to the above transient analysis technique because it is only dependant on information contained in definition of *ERL* and current transient operating point *O*.

It should be obvious to the reader that the above method is not iterative and does not require interrogation of compressor and turbine maps to determine the shaft torque. Hence the above method requires orders of magnitude lower calculation effort compared to the traditional iterative approach.

4.6 Significance of High Speed On and Off-Design Engine Models

High speed On-Design engine models make it possible to close the *Natural Design Cycle* in the On-design sense. This means that the *HD* can move through the design space in *Real Time* and can perform a human in the loop optimization of the engine cycle. Here there is no computational or functional limit on the number of design variables (i.e. dimension of the optimization problem) which can be high (e.g. 50) in a preliminary design study.

However, engine concepts are designed to satisfy performance requirements at many points within a certain flight envelope. If Off-Design calculations can also be performed at high speed (i.e. also order of milliseconds) then it will be possible for the *HD* to move through the design space in *Real Time* and perform a human in the loop optimization of the engine cycle in the *On- and Off-Design* sense. For example, Fig. 27 a) to c) show the effect of changing the design point value of TET for a turbojet (design point at SLS) from 1200°K to 1600°K. Since *CAGED* generates high speed off-design engine models, it is possible to make this movement instantaneously. In this way, the engine cycle is optimized for a mission rather than a single design point.

High speed On- and Off-Design engine models have application to engine mechanical design and weight

calculation also. The *Boeing WATE*^[46] code determines the component sizes, weights and flow path shapes using some geometrical input about each component and the most critical thermodynamic conditions (maximum temperatures and pressures) reached in the flight envelope of the engine. If high speed On- and Off- Design engine models are used, then it is possible to change the design point thermodynamic variables of the engine and determine the off-design conditions at several critical points in the flight

envelope in milliseconds. In other words, the critical thermodynamic conditions for each component (needed by *WATE*) are determined within milliseconds for any given design point. If we now couple this to *WATE*, then we obtain a thermo-mechanical design system, where thermodynamic design variables are changed, and the mechanical effect of this is observable in real time. See section 5 for more detailed description.

Another application (but not significant to *Natural Design Cycle*) of the direct off-design approach is to provide a good first estimate for larger and more accurate and sluggish programs. Often, the *Multivariate Newton Raphson*, or some other version of linear programming is used to iteratively solve the Off-Design equations. The number of iterations and success of this approach (highly non-linear functions can mislead solution) can be greatly improved if the starting point for the solution is sufficiently close to final solution.

In addition, the closed form off-design solutions of this chapter can provide a framework for those who are trying for example to obtain simple analytical formulae for off-design performance of engines, for analytical optimization of engine + aircraft. An example of this type of relation is *eq. 106* which uses design point information to calculate the whole of the *ERL*.

4.7 Earlier Work on Application of Differential Analysis to Engine Off-design Equations

After derivation of the direct non-iterative off-design method explained in the previous chapters, the author became aware of related work carried out much earlier by *Urban*^[65] and *Cockshutt*^[7].

Urban analytically derived a generic sensitivity chart by applying log derivatives to the engine off-design formulae. However he left all the equations in their raw format and did not try to analytically combine them to reach a closed form solution and the user is expected to close the equations by supplying auxiliary relations which are dependent on the type of study being carried out. For example, suppose that the user needs a relationship for the compressor exit temperature (T_{t2}) in terms of turbine inlet temperature (T_{t3}), shaft speed (U), shaft power off-take ($P_{OffTake}$), compressor inlet airflow (m_1) and burner pressure loss (π_b). With the aid of the *Urban* chart, the user can determine the following relationship,

$$\frac{\partial T_{t_2}}{T_{t_2}} = K_1 \frac{\partial T_{t_3}}{T_{t_3}} + K_2 \frac{\partial U}{U} + K_3 \frac{\partial P_{OffTake}}{P_{OffTake}} + K_4 \frac{\partial \pi_b}{\pi_b} + K_5 \frac{\partial m_1}{m_1} \quad (114)$$

where the influence coefficients (K values) come from the *Urban* chart. Now the above equation is in *open* form and the user is expected to supply auxiliary equations for the independent terms on the right hand side. So for the above example we would need to supply 5 extra equations ourselves which can come from engine control schedules, or simplifying assumptions etc.

It is very important to include control schedules/laws in the preliminary design activity of gasturbine engines. Digital controllers nowadays allow very complicated control schedules/laws to meet complicated aircraft requirements. Control schedules and limiters (e.g. an upper limit on turbine temperature) of course affect the off-design performance of the engine and hence the mechanical design of components because the components are usually sized/designed for the worst conditions reached during flight envelope or mission. The differential analysis for the off-design performance of engines provides a very convenient analytical tool for incorporating control laws / schedules as discussed in detail by *Urban*^[65]. The closed form solutions discussed in this chapter model the natural tendency/behaviour of a generic spool. A particular control schedule on an engine parameter can be included by supplying a formula/law for the corresponding log derivative term that appears in the closed form solutions. For example, a particular control schedule on bypass ratio of a variable bypass ratio turbofan could be included by supplying a formula for the ∇m_{rat} term as a function of flight mach number and altitude.

To best of author's knowledge, *Cockshutt*^[7] was the first to show the potential of differential analysis in calculating the *ERL* of engines. He derived a closed form differential solution for the *ERL* of a simple turbojet spool (i.e not a bypass spool) which compared favourably to a more rigorous computer based solution. Unfortunately he made many unnecessary simplifying assumptions and neglected many terms to make his result useful to the *CAGED* program. The *Cockshutt* result for a simple spool is,

$$\frac{\partial \pi_c}{\pi_c} = \frac{1}{1 - \frac{1}{2a_c} \frac{\tau_c}{\tau_c - 1}} \frac{\partial m_1}{m_1} \quad (115)$$

This can be compared with the restricted generic spool result derived by the author (from eq. 25),

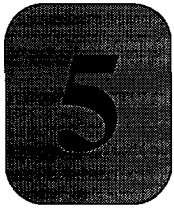
$$\nabla \pi_c = \frac{1 - \nabla \pi_b + \frac{\nabla m_{rat}}{2}}{1 - \frac{\tau_c}{(\tau_c - 1) 2 a_c}} \quad (116)$$

The reader can readily observe that the two equations have similar form. In addition, if we set the terms $\pi_b, m_{rat}, p_{t_1}, T_{t_1}$ and A_1 to constant, we obtain precisely the same result. It is important to note that this report treats the *non-restricted with recovery* (eq. 52) as well as the *restricted* generic spool. Furthermore, an additional relation was developed for the *well designed well matched* generic spool which does away with the question of restriction and can be used if minimal information is available about the type of turbine stage.

Now the comparison of the Wittenberg^{[68] [69]} method with our closed form solutions is particularly *interesting* and the reader has to be patient until the validation section 7.2.2 on page 178 where the Cocksutt, Wittenberg and the closed form solutions of this chapter are all compared together.

Application of differential methods became increasingly obsolete as faster computers and rigorous thermodynamic analysis programs became available, to the extent that there is no real mention of this approach in modern propulsion text books. The main area of their application seems to be restricted to fault diagnosis, control studies and/or guiding a rigorous iterative search for determination of *ERL*.

However, in *CAGED*, speed is a critical issue and the differential analysis, makes it possible to obtain *NDC* in the *On-design* as well as *Off-design* thermodynamic senses.



Integrated Thermodynamic and Mechanical Design Analysis Approach

The thermodynamic state of engine and performance measures are not sufficient to determine the feasibility of an engine concept. The mechanical operation of the engine, interference of components, stresses, servicing (e.g. assembly/disassembly procedure), engine geometry, weight and cost place strong constraints on the feasibility of engine concepts.

Furthermore since the engine is a geometrical and mechanical object (i.e. not just a thermodynamic network), it is highly desirable to include mechanical design aspects in the *NDC* loop alongside thermodynamic design.

Due to lack of time, the mechanical design aspects discussed in this report have not yet been programmed into *CAGED*. However all the necessary theoretical basis has been developed and the system drawings given in this chapter show the approach envisaged.

CAGED was originally intended for interactive thermodynamic and mechanical design. This is evident from the *Basic Engine Component* (Fig. 15) discussed previously and the program structure was initially developed to include mechanical as well as thermodynamic design. This chapter explains three potential possibilities of integrated thermodynamic and mechanical design in *CAGED*,

1. Engine Thermodynamics determines Engine Geometry
2. Engine Geometry Determines Engine Thermodynamics
3. Hybrid of the above two

It will appear from the following sections that the first option above is the most convenient to set up and use.

5.1 Engine Geometry Determined by Engine Thermodynamics

It may appear that the most natural approach for design of an engine is to start from engine geometry and end up with the engine thermodynamics, the second option above, since an engine is a geometrical object. However this is also the most difficult option to implement and use. Fortunately, the apparent equivalence of forward and reverse analysis in the *NDC* loop (Figure 2 on page 4) allows us to implement the easiest option (first option above), and yet obtain a very effective integrated thermodynamic and mechanic *Analysis* in the *NDC* loop.

Going from engine thermodynamics to engine geometry is much easier to implement because,

1. definition of a thermodynamic network is a much more simple task than geometric definition of components and their assembly i.e. the necessary number of variables and amount of user interaction is far less
2. All processes and modules required to set up such a system already exist and it is only a matter of time to upgrade the existing version of *CAGED* (engine thermodynamics only) to include integrated thermo-mechanical design

Fig. 28 shows the structure of the (*New*)*Object*, *Analysis* and *Properties* modules for the *NDC* which includes integrated thermodynamic and mechanical design. It is important to remember that *HD* can change any of the values contained in the *Object* module in real time by turning the appropriate dial on the workstation. The object is then fed to the *Analysis* block which in turn updates the related *Properties*. The time gap for this transformation must be of order of milliseconds for *NDC* to be possible.

The procedure starts with the design point thermodynamic variables being fed to the high speed On-Design thermodynamic model of engine (e.g. file *gt.c* as in Figure 19 on page 49). The resulting design point thermodynamic state of the model and the required thrust at this point are then input into the sizing routine which determines the mass flow through the engine and flow path cross sectional areas.

The engine is then flown through its mission/ flight envelope by successively calling the off-design engine model (e.g. file *gt.c* as in Figure 25 on page 83) at all the flight mach and altitude combinations contained in the object. Then critical conditions that each component must handle are determined, e.g. the maximum values of pressure, temperature and shaft speed that occurs during the flight envelope or mission of the aircraft.

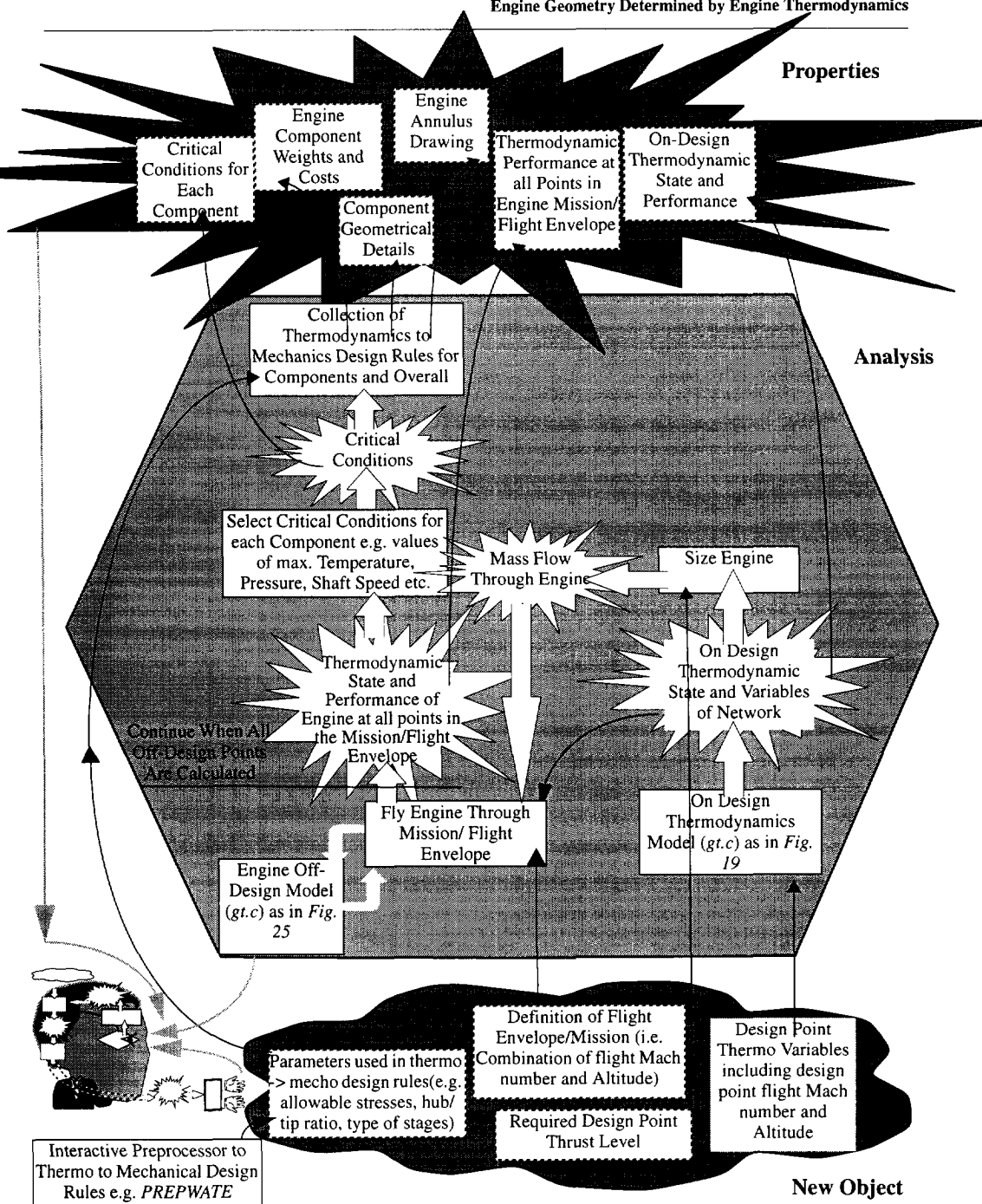


FIG. 28 Structure of Object, Analysis and Properties modules for the Integrated Thermo-mechanical Design (going from engine thermodynamics to engine geometry). Black arrows indicate information/data flow. White arrows indicate procedure and data flow. Grey arrows indicate observation paths.

FIG. 29 a) Annulus drawing of a turbofan with design bypass ratio of 2.0, weight = 12625 lbs. (figure courtesy of NASA Lewis Research Centre).

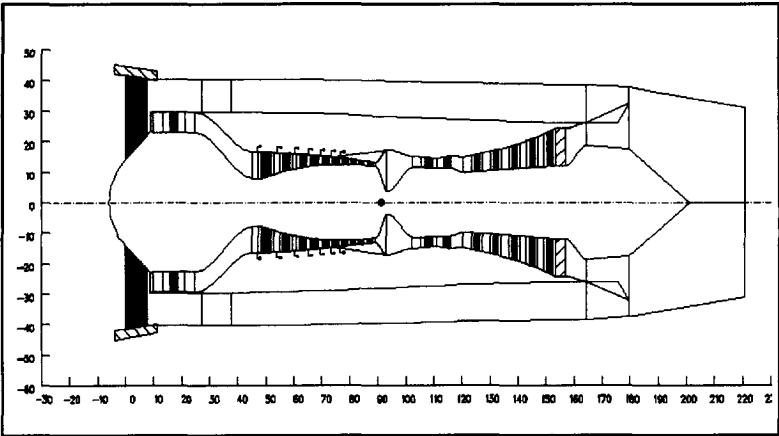


FIG. 29 b) bypass ratio = 4.0 and constant engine mass flow, weight = 9170 lbs. (figure courtesy of NASA Lewis Research Centre).

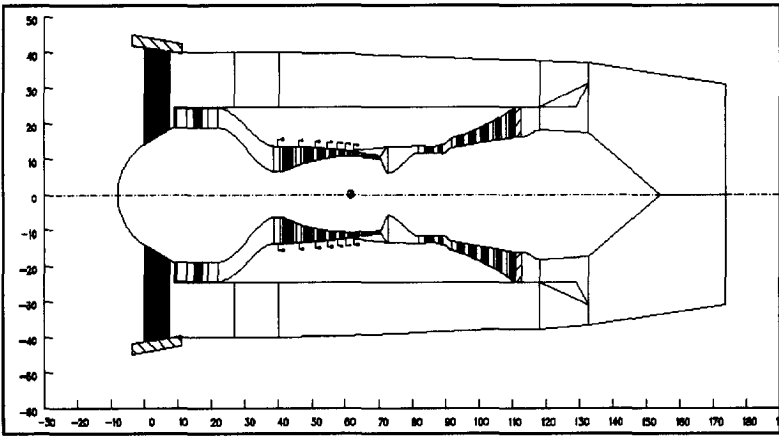
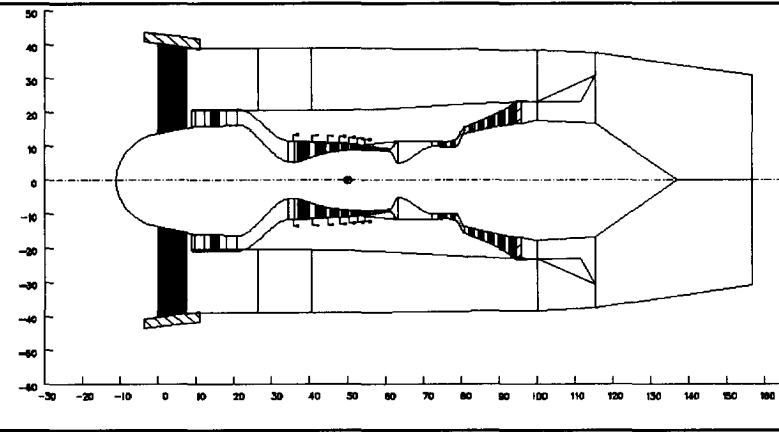


FIG. 29 c) bypass ratio = 6.0 and constant engine mass flow, weight = 7745 lbs. (figure courtesy of NASA Lewis Research Centre).



Then a collection of design rules is used to translate these critical thermodynamic conditions of each into mechanical / geometrical details for that component (e.g. *Boeing WATE* code ^[46]). The geometrical details of each component together with the *Assembly Rules* then make it possible to obtain component weight, cost (^[47]) and annulus drawing of the engine. The example *WATE* code, uses semi-analytical relationships between component mechanical design variables and component weight, which are based on analysis and component data from a number of data base engines of various configurations. For example, the following parameters are sufficient to determine compressor weight,

1. allowable pressure ratio of first stage (reflects technology level)
2. entrance and exit mach numbers
3. hub/tip ratio of first stage
4. design mode, i.e. constant mean line, constant hub or constant tip diameter
5. blade material density
6. maximum inlet and exit pressures
7. aspect ratios for the first and last stage blades
8. over-speed factor
9. blade solidity and blade taper ratio
10. compressor disc material density
11. blade volume factor

The mechanical design variables (e.g. 1 to 11 above) are contained in the (*New*)*Object* module (*Fig. 28*) and can be varied by the *HD* in real time. If the *HD* has insufficient experience/knowledge applicable to current engine design study, then the default values may be used which in turn come from a data base of previous engines, e.g. the *PREPWATE* ^[15] code.

Assembly Rules determine geometric interconnection and interference between the various engine components. The following are examples of *Assembly rules*,

1. the internal diameter of a fan cowling is a given radial distance from the fan tip diameter
2. the hp turbine diameter is not allowed to exceed the hp compressor diameter
3. the internal diameter of the hp compressor and turbine discs must be larger than the lp-shaft running through the engine

The *NDC* set up of *Fig. 28* provides a potentially very powerful engine conceptual/preliminary design tool because *HD* can make a quick variation to any of the following,

FIG. 30 a) Annulus drawing of a turbofan with design bypass ratio of 2.0, thrust = 26677 lbs SLS, weight = 7900 lbs. (figure courtesy of NASA Lewis Research Centre).

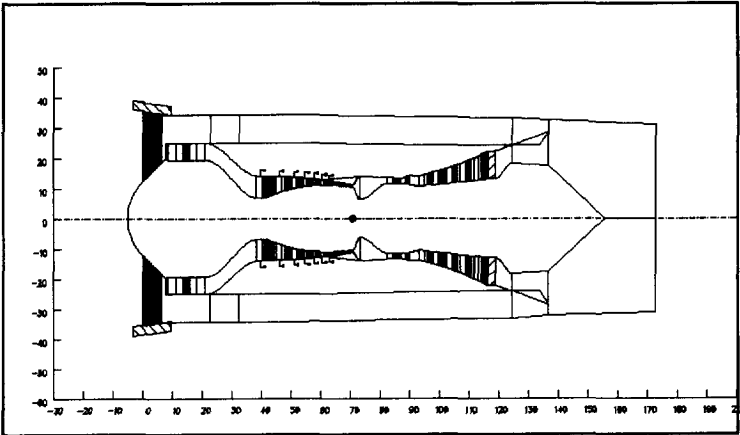


FIG. 30 b) bypass ratio = 4.0 and same thrust, weight = 9170 lbs. (figure courtesy of NASA Lewis Research Centre).

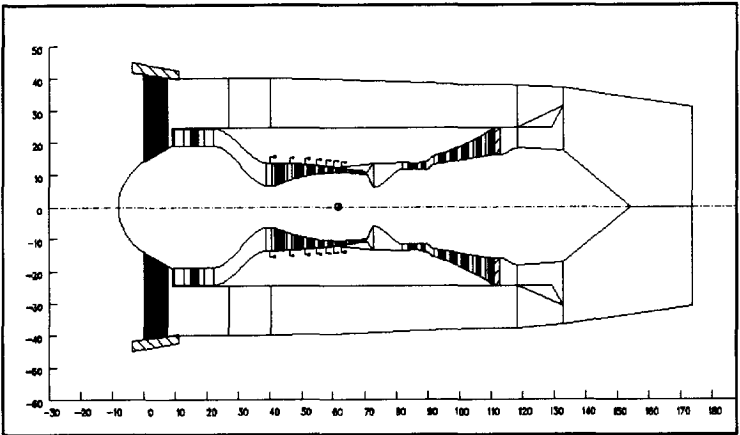
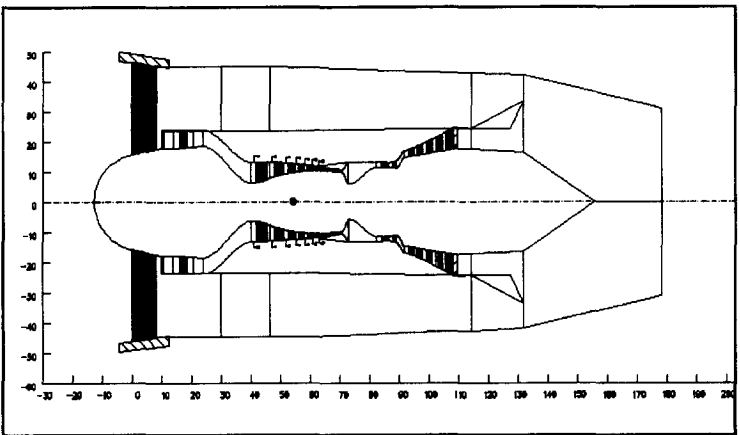


FIG. 30 c) above with design point bypass ratio = 6.0 and same thrust, weight = 10575 lbs. (figure courtesy of NASA Lewis Research Centre).



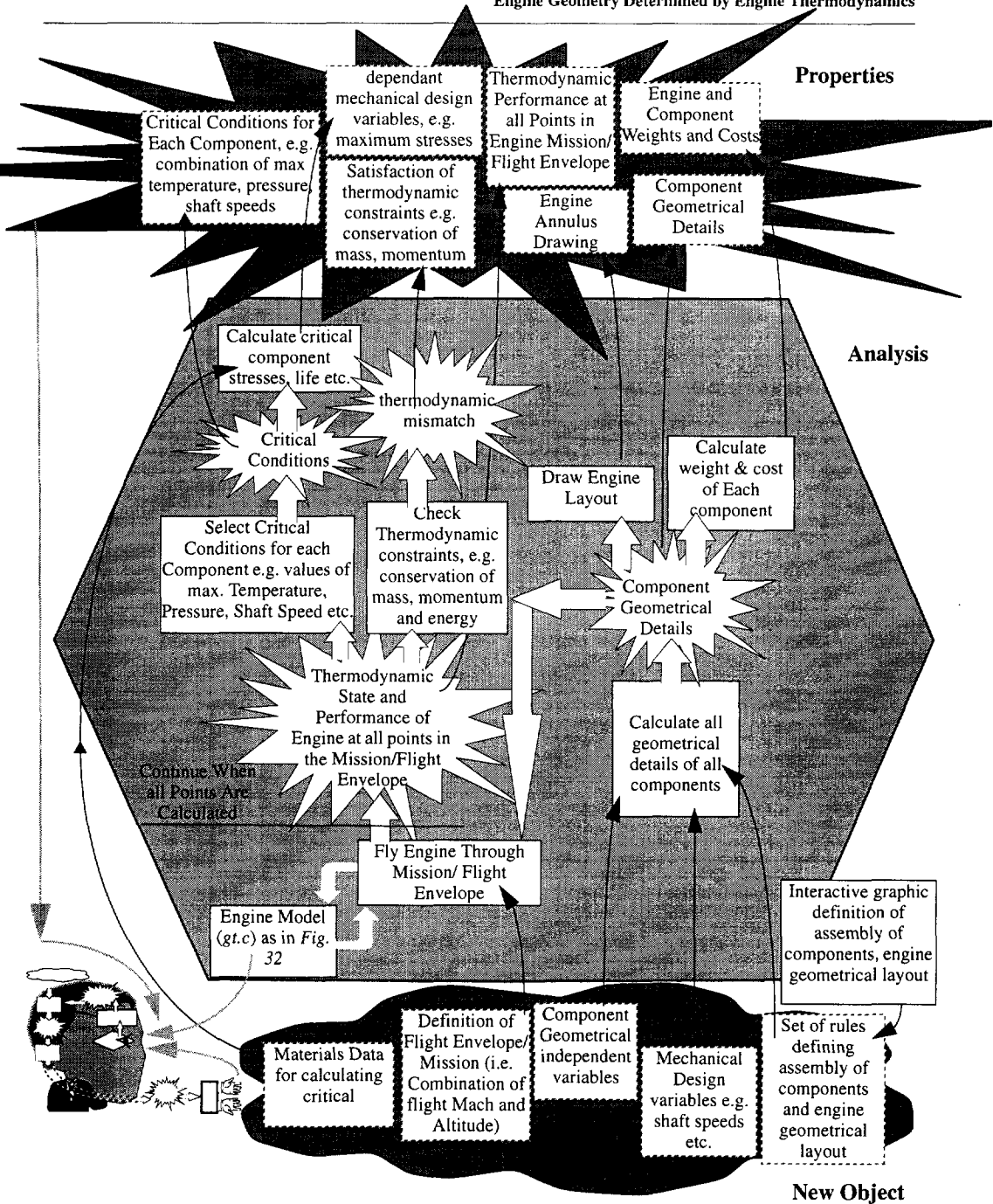


FIG. 31 Structure of Object, Analysis and Properties modules for the Integrated Mecho-Thermodynamic Design (going from engine geometry to engine thermodynamics). Black arrows indicate information/data flow. White arrows indicate procedure and data flow. Grey arrows indicate observation paths.

1. Design Point Thermodynamic Variables e.g. bypass ratio, fan pressure ratio, component efficiencies, design point flight mach number and altitude etc.
2. Required design point thrust
3. Flight Envelope or Mission e.g. pairs of flight mach number and altitude
4. Mechanical Design Variables e.g. allowable stresses, hub/tip ratios etc.

and observe instantaneously the resulting change to the following engine characteristics,

1. Engine annulus drawing
2. Engine weight and cost
3. Component geometrical details
4. Mission/flight envelope performance
5. On-design performance of the engine
6. Critical conditions for each components

Therefore the integrated approach offers the possibility to observe the mechanical consequences of any thermodynamic change to engine model in real time. This aids the user in reaching an overall thermo-mechanical optimum for the engine model. Possible mechanical problems with the current design can be avoided by observing the continuously updated engine annulus drawing. If something doesn't "look good" (e.g. if some components are interfering) then this will be reflected in the annulus drawing and the *HD* can change the parameters in the *(New)Object* module to improve the situation.

The quick execution of the *NDC* creates an apparent equivalence between forward and reverse analysis as discussed previously in section 1.1 and Fig. 2. Since the *Object* and *Properties* are observable simultaneously or in any particular order, the *HD* can consider the values contained in the *Properties* as the cause and the values contained in the *Object* as the effect. In this case the designer may get a false feeling that he is directly designing engine hardware and seeing the corresponding performance consequences.

In order to keep the execution speed of the design cycle acceptably high, *CAGED* has to generate the *ANSI C* source code, including the *(New)Object*, *Analysis* and *Properties* modules as shown in Fig. 28 where only the parameters of interest to the current study are included in the *(New)Object* and *properties* modules, consistent with modelling engine thermodynamics discussed previously.

Fig. 29 a) to c) and Fig. 30 a) to c) illustrate the power of integrated thermodynamic and mechanical design when incorporated in the *NDC*. Fig. 29 a) shows annulus

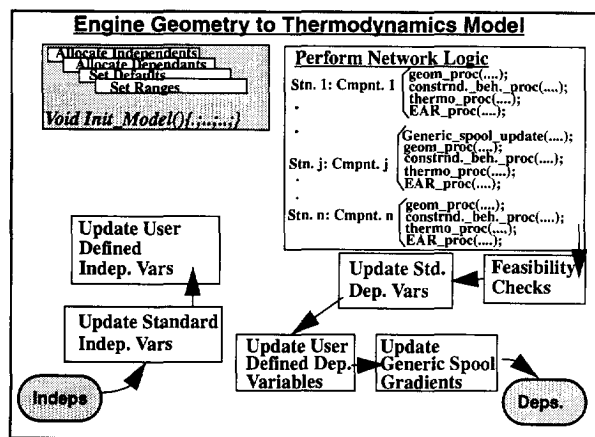


FIG. 32 The engine geometry to thermodynamics model (file "gt.c") is basically the same as the engine off design thermodynamics model (Fig. 25) except that all the geometry processes are kept in the code and there is no need to size the engine in the *Init_Model()* function because all engine geometry is known before this routine is called.

drawing of a turbofan with design point bypass ratio of 2.0 which is the result of one cycle of the *NDC* loop and which exists in the *Properties* module (Fig. 28). This integrated design system would allow the *HD* the possibility to vary the design point bypass ratio and observe the resulting change in the engine annulus drawing in real time. Fig. 29 b) & c) show the consequences of changing the design point bypass ratios to 4.0 and 6.0 respectively while keeping the engine mass flow constant. Fig. 30 a) shows a mixed flow turbofan sized for take off thrust. Fig. 30 b) & c) show the consequence of increasing bypass ratio while keeping the take-off thrust constant. When the integrated thermo-mechanical design system is implemented in *CAGED*, these changes in engine layout will appear instantaneous, thus giving the feeling that *one* is virtually designing engine in the mechanical sense.

While the *thermodynamic* \rightarrow *mechanics* design rules above enable integrated design, they may in theory carry some disadvantages with them into the system. The component statistical weight and cost correlations are derived from data base of previous engines. For example, the *WATE* method uses "volume factors"ⁱ for each component which are derived from component weight data of 29 different engines". This can place a constraintⁱⁱ on the type of engine configurations that can be studied, i.e. those engine configurations that are within or not too far from the engine data base. Also these correlations are significantly influenced by the technology level and have to

i. "volume factor" is the ratio of total to material volume of component, e.g. for compressor this is the ratio of total volume to blade volume of compressor. In this case the volume factor accounts for fir-tree mount volume, taper ratio and thickness-to-cord ratio variation.

ii. not a very serious constraint

be updated every time this level changes. Updates are necessary every time, component design technology improves.

The proposed set up of the integrated mechanical and thermodynamic design provides the *HD* with a design tool which is much more powerful than a pure thermodynamic design system. However there remains one fundamental problem. This approach makes it possible to vary component mechanical design variables without any effect on the engine thermodynamics. For example, if the designer varies the hub/tip ratio of a compressor, he/she will observe the new resulting annulus drawing of the engine in real time. But for a real compressor, this variation should result in a change in pressure ratio and efficiency of the compressor. In other words, the thermodynamic state of the compressor and the overall engine should vary as the compressor hub/tip ratio is varied. Unfortunately this variation is not taken into account in the set up of *Fig. 28*.

Therefore, if any mechanical design variable is independently varied, the program will produce an annulus drawing of the engine, which does not exactly correspond to the engine thermodynamic state and performance. Note that the reverse of this is not true, i.e. if thermodynamic design variables are independently varied, there is no mismatch between engine thermodynamics and mechanics. This mismatch is not too serious if the designer has sufficient experience and knowledge specific to the current engine design problem, because he/she will have a chance to vary the thermodynamic state of the engine in the next cycle of the *NDC* loop.

In the above example, after varying the compressor hub/tip ratio and after one cycle of the *NDC* loop is completed, the designer will vary the compressor pressure ratio and efficiency accordingly, before the next cycle begins. However it is much more desirable to be able to observe the result of any independent mechanical change directly, rather than having to make several variations to observe the result of a single variation in the state of the engine.

The *Hybrid Integrated Thermo-Mechanical Design System* (section 5.3 on page 111) attempts to remedy this fundamental problem.

5.2 Engine Thermodynamics Determined by Engine Geometry

One major disadvantage of the preceding approach is that the user has no direct influence on the geometry of the engine. There is no reason to believe that the geometry output of the *Analysis* module (in *Fig. 28*) is the only (or the optimum) geometry possible for the current combination of parameters in the (*New*)*Object*.

Ideally, an engine design cycle would start with geometrical definition of the engine and result in the corresponding thermodynamic characteristics, because the engine is a geometrical physical object (i.e. 3-dimensional) and ultimate goal of the design activity is to find this geometry that best satisfies the design requirement. Furthermore the human designer is adept at understanding and manipulating the 3-dimensional geometrical domain without requiring any additional tools or systems.

Fig. 28 shows the systematic layout of such a system. Before a source code is generated (i.e. source code including the *(New)Object*, *Analysis* and *Properties*), the *HD* (with the aid of *GUI*) defines

1. the thermodynamic network of engine components.
2. independent Geometrical Variables of engine, e.g. hub/tip ratio, internal diameter of compressor etc.
3. mechanical variables, e.g. shaft speeds
4. *assembly rules*, defining how the components are geometrically placed within the engine assembly. e.g. outer diameter of hp-turbine is equal to outer diameter of hp-compressor, inner diameter of hp-compressor disc is a given radial distance from the outer diameter of the lp-shaft etc.
5. flight envelope/ mission, i.e. combinations of flight Mach number and altitude limits
6. dependant variables/properties of interest, i.e. a selection of what is shown in the *Properties* module (Fig. 28).

Note that here the user cannot define any thermodynamic variables of any component as an independent variable because here all the component thermodynamics is a result of component geometry and mechanical design parameters. for example a compressor of given geometry, running at a given shaft speed and inlet conditions has no choice but to operate at a given pressure and temperature ratio. So the compressor pressure ratio cannot be chosen as an independent variable here.

After the above definitions, *CAGED* has enough information to generate the *ANSI C* source code for the current design study. The procedure programmed into this source code is shown in Fig. 28. The *Mechanical Design* and *Independent Component Geometry* variables are combined with *Assembly Rules* in order to give all the required component geometrical details and the engine annulus drawing. The component geometrical details can then be used to estimate engine weight and cost.

So the *HD* can vary any independent geometrical parameter (e.g. compressor hub/tip ratio, or compressor outer diameter) and observe the resulting change in engine annulus drawing immediately, before even any thermodynamic calculations have been

performed. This implies that the current geometrical state of the engine may not satisfy thermodynamic laws. For example, since the geometry of the hp-compressor is defined independently of the hp-turbine, there may be a mismatch between their respective powers.

The thermodynamic state of the engine is calculated by *Engine Geometry to Thermodynamics* model as shown in Fig. 32. This is very similar in structure to the engine off-design model (Fig. 25) except that the engine has already been sized. Another difference is that a *geometry* process is called before a *constraint behaviour* process for every component. The former provides a component performance map for the latter to use, as explained in section 4.1.1 on page 44 and Fig. 15.

Using the above *Geometry to Thermodynamics* model, the engine is “flown” through its flight envelope/mission and the critical conditions for each component (e.g. maximum temperature, pressure and shaft speeds) are recorded, as well as thermodynamic mismatches for the current geometrical state of the engine.

In order to arrive at a feasible design, the *HD* varies the geometrical state of the engine until,

1. the engine is thermodynamically feasible, i.e. conservation of mass, momentum and energy are satisfied
2. engine flight envelope/mission performance is acceptable.
3. component stresses are acceptable
4. the engine layout looks mechanically feasible, e.g. free from mechanical interference of components, etc.
5. engine weight and costs are acceptable

The advantage of this approach as compared to that of the preceding section is that the *HD* has more design freedom and can explore any geometrical engine configuration and is not dependent on a set of previously developed design rules. However the major disadvantage is that the definition of component geometry and their assembly is a very complicated task and would require much more user interaction than defining an engine thermodynamic network

Another disadvantage is that the geometrical state of engine is not necessarily thermodynamically feasible. So in order to see the end result of a geometrical change, the user has to vary other geometrical variables to find a thermodynamically feasible combination. For example, suppose the *HD* wants to see the result of increasing the hp-compressor diameter. As soon as hp-compressor diameter is increased, the hp-

compressor and hp-turbine powers fall out of balance. The *HD* has to then change hp-turbine geometry to find a matched condition for hp-spool. The change in hp-spool geometry and thermodynamic will then require geometrical change to the lp-spool and other components. The change in lp-spool geometry and thermodynamics will in turn require a change in hp-spool geometry and thermodynamics, and so on. This cycle will continue until a thermodynamical feasible condition is reached for the whole engine. Therefore, the *HD* will have to perform many geometry variations in order to obtain the end result of a single geometrical change, and, it is not clear that this inefficient process converges in reasonable time or at all.

5.3 Hybrid Integrated Thermodynamic and Mechanical Design

The foregoing discussions lead to the conclusion that going from engine thermodynamics to geometry is the simplest approach to implement and most functional. However there remains a fundamental problem that independent variations in mechanical design variables does not affect engine thermodynamics and engine performance.

The easiest way to let mechanical design variables directly influence thermodynamic design variables is by simple modification to the set up of section 5.1 on page 100. The new set-up (*Fig. 33*) make use of a *Sensitivity Module*. This module contains a set of approximate (semi)analytical relations which give the sensitivity of design point thermodynamic variables to design point mechanical design variables, for each component.

The design system operates exactly as in *Fig. 28* for any variation in the thermodynamic design variables. However if there is any variation in a mechanical design parameter, then this change is first fed to the sensitivity module. Here, the relative change in the mechanical design parameters is combined with

1. current thermodynamic state and design point variables,
2. current mechanical state and design point variables, and
3. mechanics to thermodynamics sensitivity formulae for each component,

resulting in new values of the thermodynamic design point variables. Since any independent mechanical variation results in modified values of the thermodynamic design variables, there is no mismatch between the observed annulus drawing and the thermodynamic performance of the engine. In other words, the fundamental problem of *Section 5.1* has been tackled.

However the key to the success of this approach is the availability of the above mentioned sensitivity formulae. To the author's best knowledge these do not exist anywhere. Future development of *CAGED* will have to generate these relationships in either of the following ways,

- Analysis: differentiate simple analytical formulae relating the major thermodynamic and mechanical parameters of each component.
- Data Base: compile a data table for each type of component, listing design point mechanical and thermodynamic parameters. Then derive polynomials giving the sensitivity of thermodynamic parameters to mechanical parameters.
- Use of *Artificial Intelligence* in the form of knowledge-based rules from previous experience with *CAGED*.

The author has recently received information that *NASA* has developed a method to change the design point efficiency as a function of the geometry changes and work is currently being done to extend this to off-design efficiencies also (see *NASA CR 4430* & *NASA CR 189171*).

5.3.1 Sensitivity formulae via analysis

For the majority of components, most text books give simple analytical relationships relating design point mechanical variables to design point thermodynamic variables. For example, *Cohen et. al.*^[8] give analytical relations between design point mechanical variables (i.e. blade angles, space to chord ratio, space to height ratio etc.) to design point stage isentropic efficiency.

If these relations are analytically differentiable, it is possible to analytically derive the required sensitivity relations. These sensitivities will be in terms of current design point values of thermodynamic and mechanical parameters and will be in non-dimensional form (or relative change form). Therefore the derived analytical sensitivity formulae will be used to obtain small relative perturbation in thermodynamic parameters, i.e. not for calculation of absolute values of thermodynamic parameters.

For example, consider a compressor stage. Let space to height ratio of compressor blades (ζ) and stage isentropic efficiency (η_s) be mechanical and thermodynamic design point parameters respectively. It is required to find a formula giving non-dimensional sensitivity of η_s to ζ , as a function of current design point parameters.

This sensitivity formula can be derived from simple analytical relationships for stage isentropicⁱ efficiency^[8], namely,

$$\eta_s = 1 - \frac{C_D / \left(\sigma \frac{\cos^3 \alpha_m}{\cos \alpha_1} \right)}{\frac{\Delta p_{th}}{\frac{1}{2} \rho V_1^2}} \quad (117)$$

where,

$$C_D = \text{Overall Drag Coefficient} = C_{DA} + C_{DP} + C_{DS} \quad (118)$$

$$C_{DA} = \text{Annulus Drag Coefficient} = 0.020\zeta$$

$$C_{DP} = \text{Profile Drag Coefficient} = f_n(\text{incidence})$$

$$C_{DS} = \text{Secondary Losses Drag Coefficient} = 0.018 C_L^2$$

$$\alpha_1, \alpha_2, \alpha_m = \text{Angles from velocity triangles} \quad (119)$$

$$\frac{\Delta p_{th}}{\frac{1}{2} \rho V_1^2} = 1 - \frac{\sec^2 \alpha_2}{\sec^2 \alpha_1}$$

σ = space to chord ratio

See *Cohen et.al.*^[8] for derivation of the above equations and the velocity triangles.

The ∇ operator is defined by,

$$\nabla_Y X = \frac{\partial X}{\partial Y} / \frac{\partial Y}{\partial Y} = \frac{Y \partial X}{X \partial Y} \quad (120)$$

See *eq. 11* in chapter 4 for mathematical rules regarding the ∇ operator. Applying the ∇ operation to *eq. 117* gives,

i. this is actually blade row efficiency but it can be argued that it is almost the same in value as stage isentropic efficiency^[8].

$$\nabla_{\zeta} \eta_s = -\frac{1 - \eta_s}{\eta_s} \nabla_{\zeta} C_D \quad (121)$$

Applying the ∇ operation to eq. 118,

$$\nabla_{\zeta} C_D = \frac{1}{C_D} (C_{DP} \nabla_{\zeta} C_{DP} + C_{DA} \nabla_{\zeta} C_{DA} + C_{DS} \nabla_{\zeta} C_{DS}) \quad (122)$$

But C_{DP} and C_{DS} are not functions of ζ and therefore the corresponding gradients vanish,

$$\nabla_{\zeta} C_D = \frac{C_{DA}}{C_D} \nabla_{\zeta} C_{DA} \quad (123)$$

Applying the ∇ operation to first of eq. 119,

$$\nabla_{\zeta} C_{DA} = \nabla_{\zeta} (0.020\zeta) = \nabla_{\zeta} \zeta = 1 \quad (124)$$

Substituting eq. 124 into eq. 123 and substituting for $\nabla_{\zeta} C_D$ into eq. 121 yields,

$$\nabla_{\zeta} \eta_s = -\frac{1 - \eta_s}{\eta_s} \frac{C_{DA}}{C_D} \quad (125)$$

or,

$$\boxed{\nabla_{\zeta} \eta_s = \frac{\partial \eta_s}{\eta_s} / \frac{\partial \zeta}{\zeta} = -\frac{1 - \eta_s}{\eta_s} \frac{0.020\zeta}{C_D}} \quad (126)$$

The sensitivity of η_s to σ is much more involved but can be derived in a similar manner. The sensitivity results obtained in this manner are always in terms of current values of thermodynamic and mechanical design parameters and hold for small perturbation from these current values.

Due to lack of time, the author has not attempted to analytically derive all the significant sensitivity relations for the standard components. However the above example shows that in principle it is possible to analytically derive the required sensitivities.

5.3.2 Sensitivity formulae from data base

The approach is to compile a data table for each type of component, listing design point mechanical and thermodynamic parameters. Polynomials are then derived giving the sensitivity of thermodynamic parameters to mechanical parameters, considering the major design point mechanical variable of each component as independent variables of the table, and the thermodynamic variables as dependent parameters of the table. This table is then used to derive sensitivity polynomials. The data can come from real component data, or from existing computer programs that output thermodynamic performance of components as function of geometry and mechanical design variables.

Unfortunately, the problem of availability of component data and/or computer programs to calculate them makes generating such a data base a very difficult task.

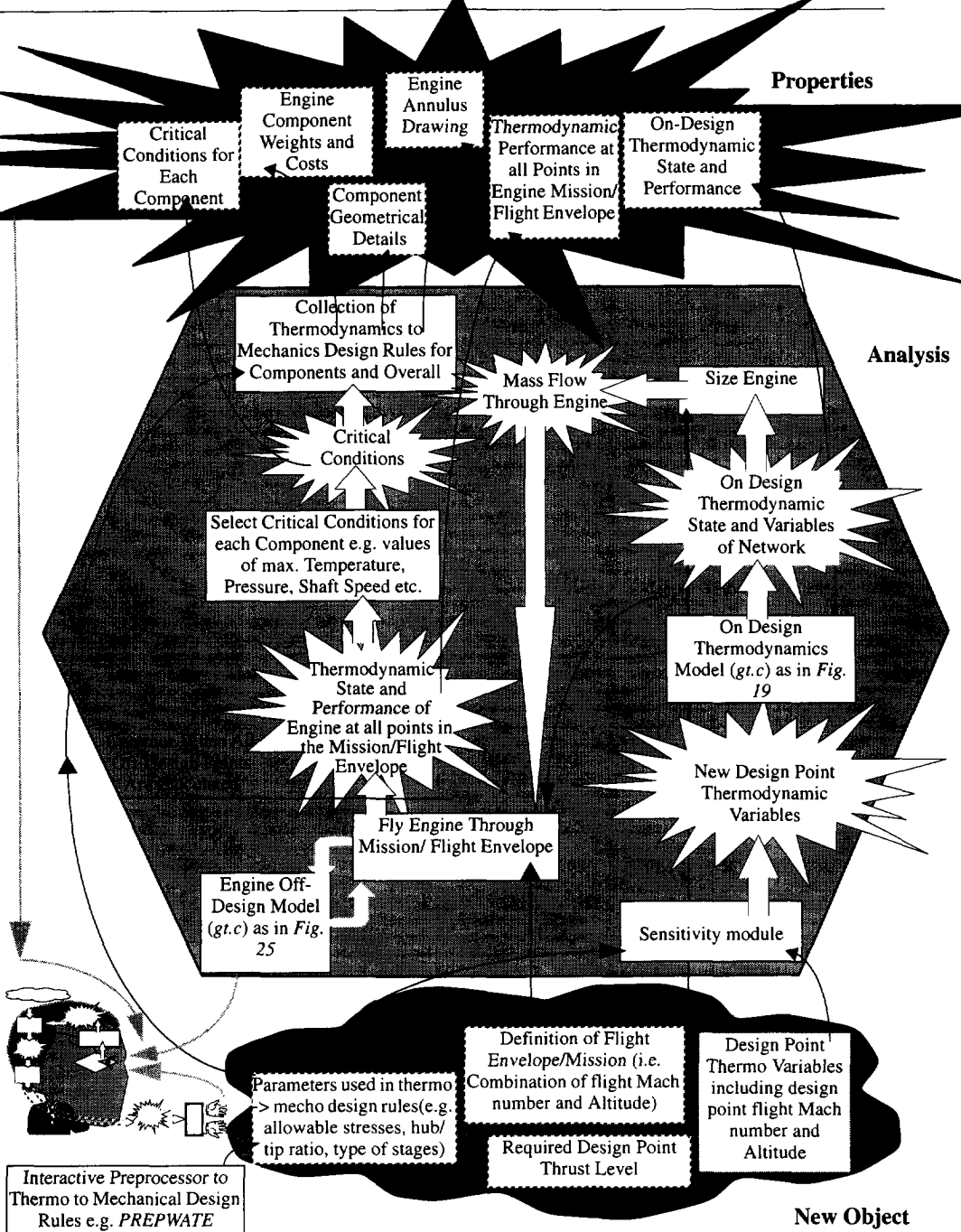


FIG. 33 The simple hybrid integrated thermodynamic and mechanic design system is basically the same as that of Fig. 28 with the addition of a "component mechanic to thermodynamic sensitivity module".

6

Numerical Optimization and Solution Methods

Despite the flexibility and advantages of including the designer within the optimization loop, there are still situations where an automatic (i.e. human out of loop) optimization is preferable. For example,

1. where the functions to be optimized do not possess patterns or trends that can be readily recognized. If optimization functions vary rapidly with discontinuities, one may not be able to make sense of a dynamic group of discontinuous and intersecting lines and curves
2. where the number of independent variables is too high to be handled by the human mind. Allocating the selection of some variables to a numerical optimization routine, allows a reduction in the unwanted high dimensions (see section 8.2 on page 223 for further discussion of this point).
3. where the *HD* accepts the disadvantages of not guiding the search strategy, only interested in the final outcome. This could be the case when a baseline design is to be selected first for later modification.

It is therefore necessary for *CAGED* to include a robust optimization algorithm. Unfortunately, the majority of optimization methods lose their applicability when solving *Highly Dimensional Functions*. The required calculation effort usually becomes enormous as dimensions increase, regardless of the elegance of the method. Most methods concern themselves with finding a single optimum which satisfies the required *Accuracy*, but which provide no quantitative measure, i.e. the *Probability* of correctness, indicating whether the true optimum is found. Furthermore, there is usually no measure of the *Calculation Effort* prior to starting the procedure.

There is a fundamental relationship (or coupling) between the *Accuracy*, *Probability* and *Calculation Effort* of an optimization method but the exact form of this relation is dependent on the algorithm used to reach the optimum. Ideally, an optimization method should facilitate the statement of required accuracy, required probability and the required calculation effort separately and the method should take care of the rest (i.e. total decoupling of the three requirements). Although this ideal case is generally not possible, it is possible to approach it by finding procedures that reduce the strength of this unwanted coupling.

This chapter derives analytical relations between the required accuracy, probability and calculation effort of a general multidimensional adaptive grid non-gradient guided search (*NGG*) method where the search points are generated either decisively or randomly. It is then shown that any adaptive method based on reducing the total solution space is heavily penalized. Furthermore, it is analytically illustrated that if the adaptive grid is randomly generated, it is far less successful than the non random adaptive grid, because the amount of grid adaptation is less decisive at every step., due to the randomness.

As with many optimization techniques, the *Dimensionality Problem* limits their application to cases where the function evaluation is executed in real time (~milliseconds) and dimensions are lower than say 25.

The present method is also particularly useful for problems in which the number of optima is known in advance. In this case the required probability can be set to its minimum value, which is required in order to distinguish an absolute optimum from a known (or likely) number of local optima. The *Coupling Relations* derived in this report will then provide the minimum calculation effort necessary to satisfy *Accuracy* and *Probability* requirements.

This chapter also pays attention to improving the *Multivariate Newton Raphson* procedure which is commonly used for function generation, optimization and solution of n by n problems (n equations in n unknowns) due to their generality and flexibility. It will be shown that a simple numerical trick, namely transforming the independent variables to indices, drastically increases the speed of the method.

6.1 Non-Random Adaptive Grid Method for Fast Optimization of Highly Dimensional, Badly Behaving Real Time Functions

Due to their computation intensive nature, *NGG* search techniques tend to be restricted to optimization of functions which are not “*Well behaved*” or for locating a significant region for further optimization by *gradient guided* methods. For highly dimensional problems however, it is not at all clear that *gradient guided* approach is advantageous regarding the *Calculation Effort* necessary to satisfy the required *Accuracy* and the required *Probability* of finding the true optimum.

The *NGG search* possesses several very attractive features as opposed to *gradient guided* search, namely,

1. the search strategy is *Function Call Intensive* rather than *Procedure Call Intensive*
2. it is possible to determine the *Calculation Effort*, which satisfies the *Accuracy* and *Probability* requirements, before starting the procedure
3. the number and complexity of *Constraint Functions* does not significantly add to calculation effort

In the present high speed modelling domain, *NGG search* techniques become very appealing because of their first quality mentioned above, i.e. the calculation time to evaluate the functions is very small compared to the calculation time to follow an elaborate procedure. Further since the relationship or the *Coupling* between *Accuracy*, *Probability* and *Calculation effort* is available, even minimal information about the merit function(F) will lead to large reduction in wastage of calculation effort. For example suppose it is known that F is not likely to have more than two optima. So a probability in excess of 50% in the solution will give us enough certainty to determine the true optimum. The *Coupling* will then provide us with the minimum calculation effort which will result in this required probability for a given error, thereby avoiding wastage.

This chapter is concerned with deriving the *Coupling* between *Accuracy*, *Probability* and *Calculation Effort* for a general *Adaptive Grid NGG* search technique where after each step, the total multidimensional space occupied by the grids is reduced in size while at the same time the grid lines are concentrated near the optimum region. The *Coupling* relations will show that for a general *merit function* (i.e. the case where F is not a special case), the search strategy is heavily penalized if the total search volume is reduced, i.e. search points must be generated in the total solution space, but they can tend to be more concentrated near the best region.

Further, the difference between *Definitive* and *Random* grid adaptation is explored, resulting in a *Break-Even* point where the *Definitive* adaptation of the grid will give better return on calculation effort.

6.1.1 Some Fundamental Distinctions

The inaccuracy or error ϵ , probability P and calculation effort E of a solution method to find optimum O are fundamentally separate but related concepts. In principle we have a multi-dimensional merit function F which is defined over a bounded search volume. The search volume is a d -dimensional hypercube of unit volume (i.e. unit edges) where each dimension represents an independent variable of the problem. The optimum O is a point (i.e. volumeless) which is situated *not* outside the search volume, where F has the best value (i.e. minimum or maximum). Or stated differently, there is no point other than O , located inside or on the boundaries of the search volume where F has a better value. A better value is a higher or lower value depending on whether we are looking for a maximum or minimum respectively.

The objective of the numerical optimization is to find a point O^+ inside or on the boundaries of the search volume such that the distance between O^+ and O is not greater than ϵ along each of the d independent dimensions. So ϵ is a measure of inaccuracy or error with which we are satisfied and is stated as a fraction of a unit interval (i.e. absolute value and not as a percentage of a particular value) throughout this chapter. Stated differently, after the optimization is done, it must be possible to fit O^+ and O within a hypercube of volume ϵ^d for us to be completely satisfied.

Now since we do not know the location of O in general (or else we would not need to optimize), we cannot directly test whether O^+ and O can be fitted into a hypercube of size ϵ^d . Therefore we need a quantitative indication or estimate of how good/true O^+ is and that is the parameter P . P is defined as the probability that O^+ and O will fit into a hypercube of size ϵ^d .

The calculation effort E is simply the amount of computation that was spent to find the result O^+ . In order to make the definition of E hardware independent, E is defined as the total number of function evaluations during the computation. This definition is very suitable here since the *Adaptive Grid* search is a *NGG* search technique based on function evaluation only.

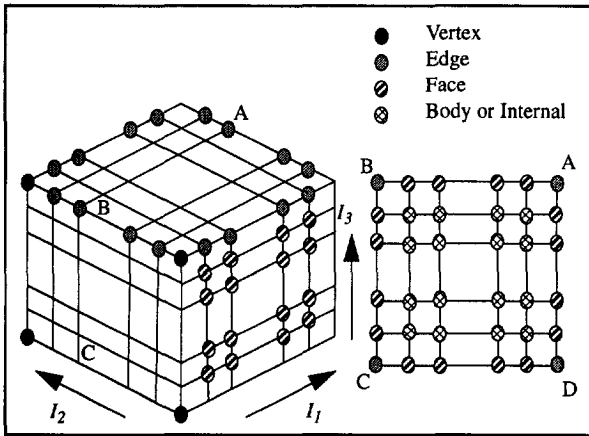


FIG. 34 Three dimensional ($d=3$) version of *The basic Grid System* with definition of Vertex, Face, Edge and Body points. The total volume of the grid system represents the *Solution Space*. The small hypercubes generated by the intersection of grid planes are called *Subregions*. The figure shows 6 grid lines in each dimension (i.e. $N = 6$).

The above four paragraphs were included so as to minimize any doubt in the readers mind as to the rigorous definition of the parameters E , P , d , F , ϵ , O and O^+ . Most practical engineers are primarily interested in a result that they can work with, and do not usually enjoy the pedantic definitions. For them the following three loosely defined points would normally be sufficient instead of the above paragraphs,

1. P indicates the probability that the found optimum is a true (i.e. global) optimum
2. ϵ is the allowable error in the final result. It indicates that if O^+ and O can be fitted together into a hypercube of size ϵ^d , then we are completely satisfied with the optimization
3. E is the computation effort spent to arrive at O^+ . For exhaustive search, this can be estimated by the number of function evaluations

This chapter derives the coupling relations between E , P and ϵ , for optimization of a general d -dimensional merit function F , where the search is done by the *Adaptive Grid* method. The derivation is for a general *Adaptive Grid* search where the total search volume reduces while the grids adapt their position within this volume, after every step. The derivation is also sensitive to whether the generation of the search points is random or deterministic.

Of course, a major parameter of interest in numerical optimization is the *Return on Investment (RoI)* which is a quantitative measure of the quality of the final result compared to the computational effort required to reach it. A simple definition could be P/E but this is not a nice meaningful number and it varies from case to case. So *RoI* does not actually appear in the derivations, but it should be clear to the reader that the

sense of the derivations is to get a measure of RoI (or a measure of P relative to E) for the adaptive grid search technique.

The *Adaptive Grid* is a *NGG* technique based on function evaluation only. This means that in practice, the search technique is suitable for the global optimization of any F that we are likely to come across in numerical optimization. But this is rather a loose definition. To be more precise condition 1 and 2 below must be satisfied:

1. F must be *piecewise continuous* inside and on the boundaries of the search volume (See reference^[6] for the definition and detailed discussion of this class of functions)
2. At least one of the left or right derivatives of F at O must have finite value

A function can be classified as piecewise continuous within an interval, even if it is not defined (or does not exist) at a finite number of discrete points within that interval. However since, the object of numerical optimization is to find the best value of F that exists this is not a problem.

The second condition above indicates the method cannot find O if it is located at an infinitely sharp needle (i.e. a double discontinuity at O). Very sharp needles do occur of course, but they are not usually infinitely sharp. In addition, to the best of the author's knowledge, there is no *NGG* numerical optimization method that can systematically (i.e. not by accident) find an O located at the point of an infinitely sharp needle.

Coupling is a quantitative relationship between E , ϵ and P which depends on the search strategy. It is possible to derive a *coupling* for the *Non-Adaptive Random* search (i.e. *simple random search*) as derived in section 6.1.3 on page 125. Equation 131 then represents the coupling for this particular search technique which is a standard result and can be found in most introductory text books^[26] on the subject. If the reader has still any doubt as to what exactly we mean by E , ϵ , P and the *coupling*, he/she should refer to the mentioned text and follow the derivation of standard result (eq. 131). This chapter attempts to derive a similar sort of *coupling* for the adaptive grid search technique.

There is in general a "dimensionality explosion" associated with *NGG* global optimization techniques, namely, the *Calculation Effort* becomes enormous as dimensions increase. For example for a 3-dimensional problem (i.e. $d=3$), $\epsilon = 0.01$, the simple random search (eq. 131) gives a very low probability $P=0.0952$ for 100000 random function evaluations. This computational effort explodes as d becomes high. Although the *Non Random Adaptive Grid* delivers many orders of magnitude better RoI

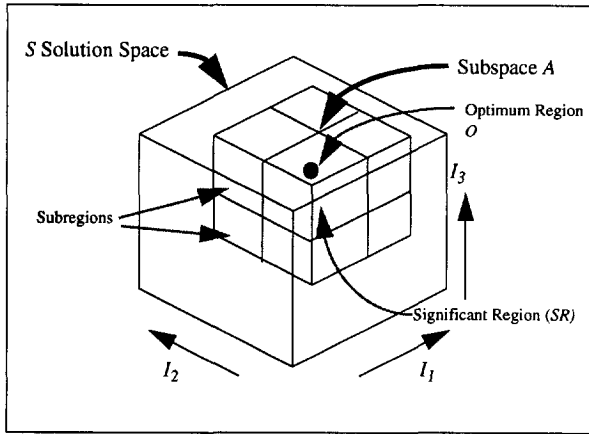


FIG. 35 Division of the solution space into the gridded region A and non-gridded region. A reduces in size by a constant factor F after every step. A is

subdivided into $(N - 1)^d$ Subregions of generally unequal size, where N is the number of grid planes cutting each dimension.

than the simple or the adaptive random search (see Figure 44 on page 139 based on the theoretical derivations of this chapter), if dimensions are high, then the application of this technique is limited to high speed or "Real Time" functions.

The *Adaptive Grid Search* is fundamentally different from the *Exhaustive Search*. The latter tries to find O directly, and is successful in finding O^+ , which is satisfactorily close to O , after many tries in the whole of the solution space. The former only tries to contain O and tries to reduce the size of the containment at every step, i.e. tries to achieve total success in the final step, hence reducing the time spent in the insignificant parts of the solution space.

6.1.2 Search Strategies

In order to provide a basis for comparison between the various methods, a universal grid system is defined which will remain the same in all methods. The difference between the methods is then found by comparing how the grids adapt their positions with each step.

Fig. 34 shows this grid and differentiates between *Vertex*, *Face*, *Edge* and *Body* points. This differentiation is necessary because the number of *Subregions* affected by various points are different. In other words, selection of different types of point as the best point, makes a different contribution towards the success of a step.

Accounting for the number of points and their Subregions is as follows,

$$\text{Number Of Points} \left\{ \begin{array}{l} N_{\text{Grid Points}} = N^d \\ N_{\text{Vertex}} = 2^d \\ N_{\text{Edge}} = d2^{d-1} (N-2) \\ N_{\text{Body}} = (N-2)^d \\ N_{\text{Face}} = N_{\text{Grid Points}} - N_{\text{Vertex}} - N_{\text{Edge}} - N_{\text{Body}} \\ = N^d - 2^d - d2^{d-1} (N-2) - (N-2)^d \end{array} \right\} \quad (127)$$

$$\text{Number Of Subregions Affected} \left\{ \begin{array}{l} N_{\text{Subregions}_{\text{Vertex}}} = 1 \\ N_{\text{Subregions}_{\text{Edge}}} = 2 \\ N_{\text{Subregions}_{\text{Face}}} = 2^{d-1} \\ N_{\text{Subregions}_{\text{Body}}} = 2^d \end{array} \right\} \quad (128)$$

Where $N_{\text{Subregions}_X}$ signifies the number of *Subregions* affected by points of type X .

In fact it is possible to distinguish d types of points instead of only the four types above. Apart from the simplicity in the final result, using only four types of points will offset the underestimating nature of the final result. See section 6.4.3 on page 150 for a more rigorous derivation using d types of points.

The following sections will derive the coupling relations for the Adaptive and Non-Adaptive *NGG* search strategies. Recognizing that the required computational effort varies between random and deterministic grid generation, for convenience in comparison, the calculation effort will be estimated by the number of function calls only and grid calculation efforts will be neglected. So,

$$E_{\text{All Methods}} = N^d \times N_s \quad (129)$$

It will be shown that reducing the search volume is very counter productive regarding *RoI*. If the search volume is constant, then the bounding grids are not moving, and the

vertex points and the edges remain fixed in position. Also to delay dimensionality explosion, the merit function is not evaluated at face and edge points, but calculated from values at vertex points, by interpolation. Hence the calculation effort becomes,

$$\begin{aligned} E_{\text{Constant Volume}} &= 2^d + N_{\text{Body}} N_s \\ &= 2^d + (N - 2)^d N_s \end{aligned} \quad (130)$$

6.1.3 The Non-Adaptive Random Search

Here all the N^d search points are generated randomly and it is hoped that *at least one* of the search points (say O^+) will succeed by itself. O^+ will succeed if its position is such that O and O^+ can fit into a hypercube of size ϵ^d . In this method the information that comes from which of the search points so far has the best F value is not used in any way in the generation of the following search points, i.e. the position of the search points are totally independent from each other. The probability that one independent search point will directly succeed is ϵ^d . Therefore probability of a search point failing is $1 - \epsilon^d$. The probability that $N^d \times N_s$ points will fail is then $(1 - \epsilon^d)^{N^d \times N_s}$, and hence,

$$P_{\text{Non-Adaptive Random}} = 1 - (1 - \epsilon^d)^{N^d \times N_s} \quad (131)$$

where the number of search points has been chosen as $N^d \times N_s$ to facilitate future comparison with adaptive grid methods. This is a well known result, which shows that in the limit, as the number of search points tends to infinity, P tends to unity.

6.1.4 The Non-Random Adaptive Grid Search

6.1.4.1 The Basic Search Strategy

The underlying principles are very simple. A subspace A of solution space S is selected (Fig. 35). A itself is subdivided into $(N - 1)^d$ subregions by N grid planes cutting each of the d dimensions. The strategy is as follows,

1. Start with A identical to S .

2. Grid the space enclosed by A . The initial order of the grid planes need not be regular i.e. the Subregions of A do not have to be equal in size to begin with.
3. Evaluate F at all grid points and select the best point. This implies that we have selected a number of best Subregions that surround this point. The number of Subregions affected depends on the type of the best point as given in eq. 128
4. Adapt the position of the grid planes towards this best point. This implies that after grid adaptation, both the volume of A and of the affected Subregions are reducing.
5. Repeat steps 3 & 4 until the grid planes are so close together that the size of the best Subregions is equal or less than ϵ^d . The Optimum is now found to within an error radius of ϵ .

Note: in step 5, if the search volume is constant (i.e. vertex points and edges are fixed in space) and if d is high, then in order to delay the dimensionality explosion, re-evaluate the function only at body points. The rest of the points can then be evaluated by interpolation using the function values at body and vertex points.

6.1.4.2 The Coupling Relations for the Non-Random Adaptive Grid Method

Again, the overall search or optimization is successful if the resulting best point O^+ and the actual optimum O , can be fitted into a hypercube of size ϵ^d . In order to find an expression for the probability (P) of this overall success for the *Adaptive Grid Search*, we need to find a robust definition for *Stepwise Success* of each adaptation step. Our definition will be robust if on the basis of this definition, the probability P_{step} of the *Stepwise Success*, of different steps, contribute *independently* towards the probability of overall success. A robust definition for *Step Success*, would therefore imply that the success of one step does not affect the success of the following steps. It is initially difficult to accept, that a definition can be found such that the step success of successive steps do not influence each other. This difficulty arises due to the fact that the natural way to think about step success is that “a step is successful if a search points falls close to the optimum point O ”. However this is a “greedy” way of looking at the problem. The objective of the adaptive grid is to make moves so that a good result is obtained after the final step, and not directly.

Consider the following definition for *Step Success*,

- *Def1*: A step is successful if and only if the optimum O is located in one of the affected Subregions of the selected best point **and**

remains there after these Subregions have been shrunk by the grid adaptation.

Accepting *Def1* for a moment, it can be seen that if a step is *successful*, then this will lead to *success* in the following step, if the shrinkage is small. Or in other words, success of one step can contribute to the success of the next step.

The following definition of stepwise success has ultimate robustness,

- *Def2*: A step is successful if the selected best point hits one of the *Significant Regions (SR)* of the function to be optimized. *Significant Regions* are locations in the search space, which will drive the search strategy to the true optimum, by the time the final step is completed. A *SR* can be of any shape or size and can be located anywhere in the search space. The shape, size and location of a *SR* is not fixed and can change between successive steps and is dependent on the merit function *F* and the current position of the grids.

In other words, *Def2* says that if a step is successful (i.e. hit a *SR*), then the true optimum will be found after the final step is completed (note: not vice versa). *Def2* may appear trivial but it is necessary in deriving the *Coupling* relations. See section 6.4 on page 149 for clarification of *Def2* and step success. It is easy to see that if we use *Def2*, the stepwise success of a particular step does not necessarily affect the stepwise success of the following steps.

Let P_{step} represent the probability of success in the context of *Def2*. The overall search fails if step 1 and step 2 and... and step N_s fail. If we find a good estimate for P_{step} and consider each step to be an independent *Bernouli Trial* then we can use the *Binomial Distribution* and the following logic to estimate probability of success after completion of the last step,

$$\begin{aligned} \therefore P_{\text{Adaptive Grid}} &= 1 - \text{Prob}(\text{Zero Success After } N_s \text{ Steps}) \\ &= 1 - (1 - P_{step})^{N_s} \end{aligned} \quad (132)$$

If P_{step} is not constant in every step then,

$$P_{\text{Adaptive Grid}} = 1 - \left(1 - P_{step_1}\right) \times \left(1 - P_{step_2}\right) \times \dots \times \left(1 - P_{step_{N_s}}\right) \quad (133)$$

FIG. 36 Concentration of grid lines near a Favoured Point. An imaginary, straight and weightless flexible beam is used along each independent direction in the solution space in order to define the relation between old and new grid positions. At a favoured point where it is required that the grid lines be closer together, a couple is applied to the beam. The resulting deflection of the beam is immediately calculated from the imaginary “beam equations”. The deflected beam is then used to get new grid positions.

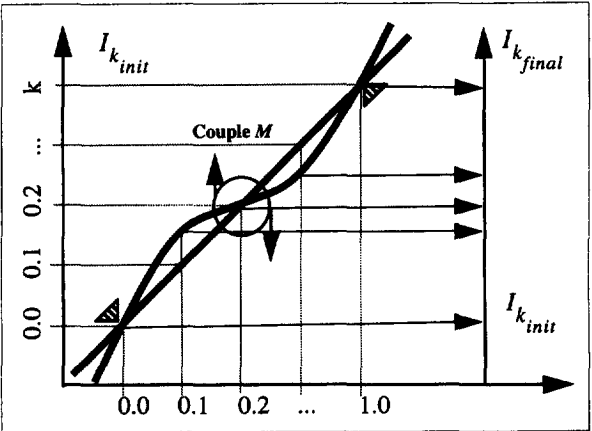


FIG. 37 Determination of Couple M_{max} from the required final accuracy. M_{max} is the couple that twists the beam such that the slope of the beam is $(\Delta I_{k_{final}}) / (\Delta I_{k_{init}})$.

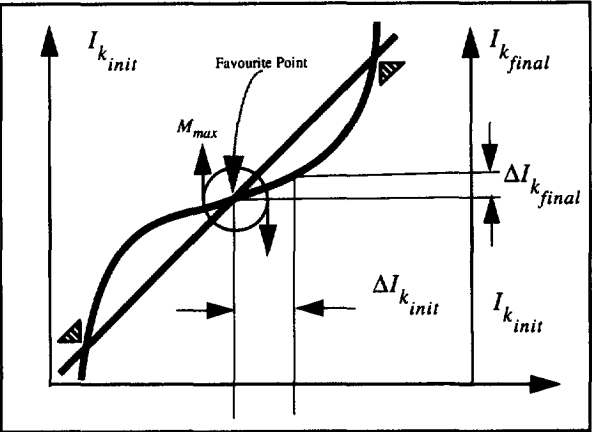
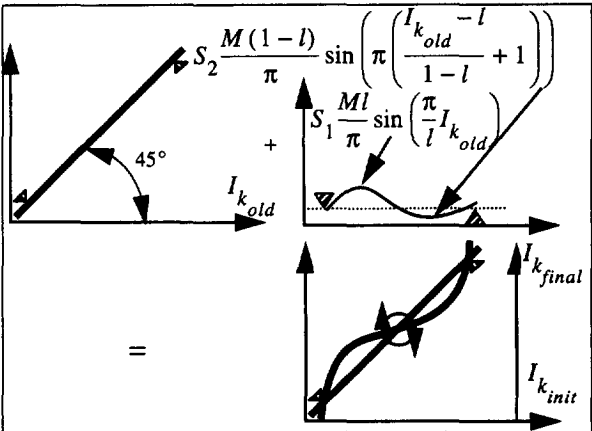


FIG. 38 The “beam equation” is the addition of two sinusoidal curves passing through favoured point and a straight line of slope 1.



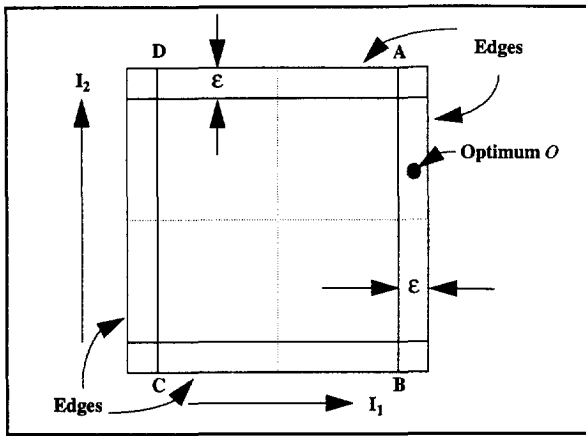


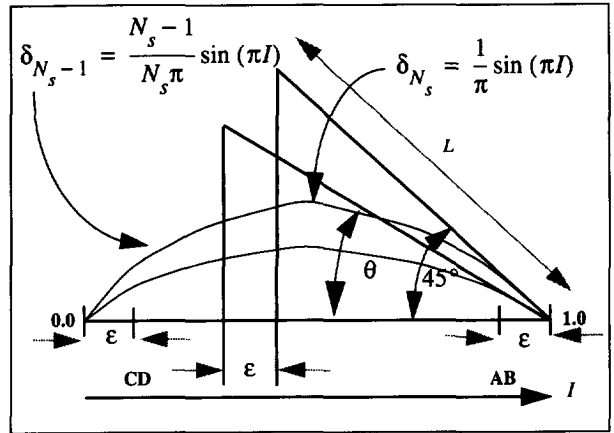
FIG. 39 A 2d view of the minimum gridding. The starting position of the gridding is significant. Suppose optimum O falls between the edge and the preceding grid line. If the best point is chosen on the edge grid of I_1 , then the preceding grid line can adapt and move towards O . If the best point is chosen on the line AB , then AB and the edge cannot get closer due to the "beam equations". Therefore the grid line AB is initially placed a distance ϵ from the edge.

where $P_{\text{Adaptive Grid}}$ represents the probability of overall success of the whole optimization strategy. The problem that now remains is how to test for the conditions of *Def2* for every step. Since no priory information is available about F and O , due to the way in which *Significant Regions* are defined, it is actually impossible to measure accurately, in each step, the probability of step success for each step. However we can make a conservative estimate of this probability by realizing that if the conditions of *Def1* are satisfied for a given step, then this *definitely* contributes to the finding the optimum O in the final step.

In other words it was necessary to use *Def2* to derive the *Coupling* relation (eq. 133). However in order to get a measure for P_{step} , P_{step} is estimated (on the low side) by the probability that the optimum will fall in one of the affected subregions of the selected best point B and will remain there after the subregions shrink (i.e. the conditions of *Def1*). In this way $P_{\text{Adaptive Grid}}$ will be underestimated by eq. 132 or 133.

Now several events are necessary to satisfy *Def1*. In the first place O must fall into gridded subspace A . Assuming that after each step the volume of A is reduced by a constant factor f , then at step J , the size of A is given by f^J , and the probability that O will fall in A is then $\text{Prob}(O \text{ falls in } A) = f^J$. Further B can be either a *Vertex*, *Edge*, *Body* or a *Face* point (Fig. 35). The number of affected regions (i.e. the Subregions that immediately surround B) will be different according to the type of B as given by eq. 128. Since the outcome of the selection of B are *mutually exclusive* events (e.g. B cannot be both *Vertex* and *Body*), then the probability of success will be the addition of the individual probabilities for each type of B . If O falls in A and in the affected Subregions of B then the probability that it remains in the affected Subregions is K ,

FIG. 40 Estimation of maximum and minimum N_s by considering the final adaptation step. Shown here is the beam deflection at the final and the prefinal step. $N_{S_{max}}$ is the value of N_S above which all the mobile grid planes have converged into an interval of width ϵ . $N_{S_{min}}$ is the minimum value of N_S which ensures that the difference between the two δ is not too large.



where K is the shrinkage factor of the affected Subregions (assumed to be constant). Collecting all the events that leads to a successful step gives for general step J ,

$$\begin{aligned}
 P_{step_j} &= Prob(O \text{ falls in } A) \times \\
 &\times \sum_{i=Vertex}^{face} \left(\begin{array}{c} Prob(\text{Point of type } i \text{ will be selected}) \\ \times \\ Prob(O \text{ will be in one of the affected subregions of } B) \end{array} \right) \times (134) \\
 &\times Prob(O \text{ remains in affected subregions of } B \text{ after shrinkage})
 \end{aligned}$$

Or fully written out,

$$\begin{aligned}
 P_{step_j} &= f^j \times \left(\begin{array}{c} \frac{N_{Vertex}}{N^d} \times \frac{N_{Subregions_{Vertex}}}{(N-1)^d} + \frac{N_{Edge}}{N^d} \times \frac{N_{Subregions_{Edge}}}{(N-1)^d} \\ + \\ \frac{N_{Face}}{N^d} \times \frac{N_{Subregions_{Face}}}{(N-1)^d} + \frac{N_{Body}}{N^d} \times \frac{N_{Subregions_{Body}}}{(N-1)^d} \end{array} \right) \times K \\
 &P_{Grid}
 \end{aligned} \quad (135)$$

K is the shrinkage factor of the Subregions per step and can be estimated if it is assumed to remain constant. At the start of the search, the size of A is unity (i.e. maximum). The

size of $(N-1)^d$ Subregions of A are then on average $\frac{1}{(N-1)^d}$. After the final step is completed, the best Subregion will have size ϵ^d . If the shrinkage factor remains constant in each of the N_s steps then,

$$K = (\epsilon(N-1))^{\frac{d}{N_s}} \quad (136)$$

Manipulating eq. 135 with the aid of eq. 127 & 128 gives,

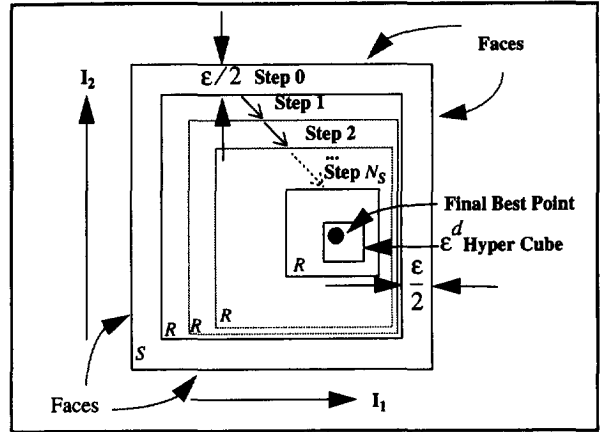
$$P_{step_J} = f^J \times K \times \left(\frac{2^d (1 - 2^{d-1}) + d 2^d (N-2) (1 - 2^{d-2}) + 2^{d-1} (N-2)^d + 2^{d-1} N^d}{N^d (N-1)^d} \right) \quad (137)$$

Equations 129, 133, 136 & 137 fully define the *Coupling* for a general adaptive grid search where the gridded search volume is reducing in size and the best Subregions are shrunk. Variations on the search method are produced by setting particular values to the three terms on the RHS of eq. 137. For example,

- Setting $K=1$ represents a shrinking A method where the best Subregions are not given bias
- When $N=2$, the middle term becomes unity. Setting $K=1$ and $N=2$ represents a method where the search volume (which is not subdivided into Subregions) is shrinking
- Setting $f=1$, represents a method where the search volume remains as large as the total solution space.

For $N = 5, d = 5, \epsilon = 0.01, f = 0.9, N_s = 40$, the above relations yield $K = 0.668, P_{grid} = 0.01565$. The first term is the contribution of reducing the size of the search volume A . Like all the three terms its effect is to reduce the success of the each step, however it becomes more and more penalizing as the search continues (i.e. as the value of J increases). In the example case, although the starting value of f is quite high, after 30 steps its magnitude reduces to 0.0423. The increasing penalization then accumulates, resulting in very low overall probability of success for a given N_s and N . The only way to counteract this effect is to use a value of f close to unity, implying that the search points must be generated in all of the solution space in each step. Therefore,

FIG. 41 In the *Random Adaptive Grid* the search points are initially distributed equally in the solution space S . But as the search goes on, the distribution of the search points becomes more and more concentrated in some area. This can be visualized by imagining the maximum likelihood hypercube R to be reducing in size as shown. R is the region where search points are $(1 - \epsilon)^d$ likely to fall.



- Any NGG search technique based on *Reducing The Search Volume* has a very low RoI!

The above conclusion, however, does not hinder the possibility of increasing the concentration of search points in a particular region, as long as enough search points are generated to cover the total solution space.

The middle term of eq. 137 is the contribution of the gridding to the stepwise success, P_{Grid} . Evaluating P_{Grid} for various values of d and N , gives the perhaps surprising result that P_{Grid} increases as the number of grid lines N decreases!. The reason is that the adaptive grid tries to find the optimum in the final step and not immediately in one go. The less the grid lines, the larger the affected subregions of the selected best point, and therefore the smaller the chance that the selected best point will miss the SR .

Setting f to unity in eq. 137 makes the step success constant for every step. Using this in eq. 132 gives measure of the overall success for the general adaptive grid method,

(138)

$$P_{\text{Adaptive Grid}} = 1 - \left[1 + \frac{\left(2^d (1 - 2^{d-1}) + d 2^d (N-2) (1 - 2^{d-2}) + 2^{d-1} (N-2)^d + 2^{d-1} N^d \right)}{(N^d (N-1)^d)} \times \left(\epsilon (N-1) \right)^{\frac{d}{N_S}} \right]^{N_S}$$

Gridding P_{grid}

Shrinkage K

This coupling actually underestimates the probability of overall success of the search because conditions stated in *Def1* were used to estimate the conditions of *Def2*.

If the search volume is not reducing, then the bounding grids are not moving, and the edge and vertex points remain fixed in position. Also to reduce dimensionality explosion, the function is not evaluated at face and edge points, but calculated from values at vertex points. Hence the calculation effort becomes,

$$E_{\text{Constant Volume}} = N_{\text{Vertex}} + N_{\text{Body}} N_S$$

$$= 2^d + (N-2)^d N_S$$
(139)

The above gives the *coupling* for a *Non-Random Adaptive Grid NGG* search method, where the grid is decisively adapted at a constant rate. If the grid is randomly generated (rather than decisively) then there is no guarantee that the required shrinkage will be achieved, and more steps will be needed to reach the solution. See next section.

6.1.5 Grid Adaptation Using Imaginary Flexible Beam

For a given grid system and calculation effort (i.e. given N , d & N_S), the only way to influence the overall success is through the shrinkage factor K . For simplicity in derivation of eq. 138, K was assumed to remain constant with steps. The resulting *Coupling* relation 138 is then independent of the actual way in which the grids are adapted.

Considering *Def2*, it can be seen that if a single step is successful (i.e. probability that a search point hits a *SR* is 100%), then it is **guaranteed** that the search will lead to the true optimum **after** all the following steps are completed. To achieve 100% probability of step success is not realistic, but the reader can appreciate that if we achieve higher levels of probability of step success, in the beginning stages of the search, then the burden lessens on the steps that follow. For a given N , N_s and d , this implies that the grid adaptation must be relatively slow in the early steps of the optimization. In addition, during the final stages of the search, relatively high levels of probability of step success are required, because there are then very few steps left to correct any wrongdoing of the previous steps. Therefore, to get the best return (i.e. probability of overall success) on investment (i.e. calculation effort), we conclude the following,

- Given N_s , the *Grid Adaptation* should be relatively *Slow* during the *Early* and *Final* steps of the search. It can then be relatively *Faster* in the *Intermediate* stages to make up for lost time.

The method of grid adaptation determines therefore how efficiently the calculation effort is being used. This report describes a grid adaptation where an imaginary flexible beamⁱ is used to move the grid lines in each dimension towards a favoured region (i.e. Towards the current best point). *Fig. 36* shows this beam which in its undeflected state is a straight line with slope of unity. The horizontal and the vertical axis represent the old and new position of grid lines respectively. A couple M is incrementally applied at the current best point and the beam is deflected. The shape of the deflected beam then projects the old position of grid planes into their respective new positions. The magnitude of couple M is increased by equal amounts at every step until finally after N_s steps its magnitude is such that the final grid spacing at the best point is ϵ .

If I_k represents the dimensionless independent variable along dimension k , which can take values in the interval $0.0 \leq I_k \leq 1.0$, then the transformation between new and old values due to the deflection is given by,

$$I_{k_{new}} = I_{k_{old}} + S_1 \frac{Ml}{\pi} \sin \left(\frac{\pi}{l} I_{k_{old}} \right) + S_2 \frac{M(1-l)}{\pi} \sin \left(\pi \left(\frac{I_{k_{old}} - l}{1-l} + 1 \right) \right) \quad (140)$$

i. this imaginary flexible beam also has an imaginary "beam equations"

where l represents the position of the best point along the beam and M the magnitude of the couple which is applied at the current best point. S_1 and S_2 are switches such that,

$$S_1 = \begin{cases} 1, & \text{If } (0.0 \leq I_{k_{old}} \leq l) \\ 0, & \text{If } (l \leq I_{k_{old}} \leq 1.0) \end{cases} \quad (141)$$

$$S_2 = !S_1$$

Where “!” is the logical *Not* operator (i.e. if $S_1=1$ then $S_2=0$ and vice versa) and *eq. 140* and *141* are visualized in *Fig. 38*.

The maximum possible value of M is unity. Beyond this value the deflected beam will have negative slope. The maximum required couple M_{max} is the value of the couple which results in a spacing of ε between the grid lines as shown in *Fig. 37*, namely

$$M_{max} = 1 - \frac{\Delta I_{k_{final}}}{\Delta I_{k_{init}}} = 1 - \varepsilon (N - 1) \quad (142)$$

Now the couple M is incrementally applied until after N_s steps, $M=M_{max}$. The couple increment is then,

$$\Delta M = \frac{1 - \varepsilon (N - 1)}{N_s} \quad (143)$$

Substituting $M = J \times \Delta M$, $J = 1, 2, \dots, N_s$ into *eq. 140* gives the grid adaptation equation (or the “beam equation”) for the J^{th} step and the k^{th} dimension,

(144)

$$I_{k_{new}} = I_{k_{old}} + J \frac{1 - \varepsilon (N - 1)}{\pi N_s} \left[S_1 l \sin \left(\frac{\pi}{l} I_{k_{old}} \right) + S_2 (1 - l) \sin \left\{ \pi \left(\frac{I_{k_{old}} - l}{1 - l} + 1 \right) \right\} \right]$$

It is relatively easy to visualize that the grid adaptation is slow in the early stages due to small values of J (i.e. small amplitude of the sinusoidal functions). As the magnitude of the couple increases, the beam deflection increases and the grid lines take larger and larger steps towards the best point. However as the grid lines approach the proximity of the beam inflection, then the difference between the deflected and the undeflected beam becomes increasingly less significant, thereby resulting in slow adaptation near the best point. The “beam equations” therefore satisfy the general requirements for grid adaptation discussed above.

6.1.6 Practical Issues

Basically a choice of N and N_s must be made such that the calculation effort E will be a minimum to satisfy the required *Probability* and *Accuracy* for optimization of a d dimensional merit function (F). Equation 129 suggests that for large d , the calculation effort is extremely sensitive to N as opposed to N_s .

Given (N, ϵ, d) , N_s can be increased in order to increase P (eq. 138). However there is a maximum meaningful value of N_s beyond which all grid lines have converged and cannot converge further, thereby limiting the maximum achievable P . Stated simply,

- Minimum Calculation Effort is achieved by choosing minimum N , as long as the necessary N_s remains meaningful.

Further there is a minimum value for N which is still meaningful. If the total search volume is to remain constant then we need two grid lines at $I_k = 0$ and $I_k = 1$ along each dimension k which will not move. Then at least two more grid lines are necessary to contain the optimum in each dimension. So the basic minimum number of grid lines are necessary to give adaptation meaning is,

$$\boxed{N_{min} = 4} \quad (145)$$

For low values of d (~ 5) it is possible to use higher N than this value, and still get an acceptable calculation effort.

The starting position of the grid lines are also important. Initially a grid line is placed at a distance of ϵ from each edge, to avoid grid adaptation failure in case O is very close to the edge as shown in Fig. 39.

After $N_{S_{max}}$ steps, all grid planes (except the edges) converge into one region of volume ϵ^d . So any more grid adaptation after $N_{S_{max}}$ steps has no meaning. In the worst case, the optimum sits at one of the vertices of the solution space and all the grid lines have to converge to one corner. Considering this in one dimension (*Fig. 39*), the furthest grid plane CD has to move all the way to the opposite end until it passes the initial position of AB . Further, there is a lower limit on N_S below which the grid adaptation is too fast in the beginning and final steps, i.e. $N_{S_{min}} < N_S < N_{S_{max}}$.

Simplifying *eq. 144* for this special case, $S_2 = 0$, and $l = 1$,

$$\delta = \frac{J}{N_s} \frac{1}{\pi} \sin(\pi I) \quad (146)$$

where δ denotes the movement in the position of a grid line positioned at I . Now considering the two final steps,

$$\delta_{N_s-1} = \frac{N_s-1}{N_s} \frac{1}{\pi} \sin(\pi I), \delta_{N_s} = \frac{1}{\pi} \sin(\pi I) \quad (147)$$

Fig. 40 shows the δ curves of this equation. As N_S increases, the difference between the two δ curves becomes smaller. $N_{S_{min}}$ is the value of N_S above which the difference between the slopes of the two delta curves at the extreme end is acceptably small. In other words,

$$L \cos \theta - L \cos 45 = \epsilon \quad (148)$$

From *eq. 147* the slope of δ_{N_s-1} is,

$$\tan \theta = \frac{N_s-1}{N_s} \Rightarrow \cos \theta = \frac{N_s}{\sqrt{2N_s^2 - 2N_s + 1}} \quad (149)$$

Combining equations 148 & 149 and solving for N_S in the resulting quadratic gives,

$$N_s = \frac{1 + \sqrt{1-H}}{2H} \text{ where } H = 2 \times \left(1 - \frac{1}{\left(1 + \frac{\sqrt{2}\epsilon}{L} \right)^2} \right) \quad (150)$$

where the larger root is meaningful. L is the length along which δ_{N_s} is approximated by the -45° line as shown by Fig. 40, i.e.

$$\frac{1}{\pi} \sin\left(\pi \frac{L}{\sqrt{2}}\right) = \frac{L}{\sqrt{2}} - \epsilon \quad (151)$$

Expanding the LHS up to and including the cubed term and solving for L , we find:

$$\begin{aligned} \frac{L}{\sqrt{2}} - \epsilon &= \frac{1}{\pi} \left(\pi \frac{L}{\sqrt{2}} - \frac{\left(\pi \frac{L}{\sqrt{2}}\right)^3}{3!} \right) \\ \rightarrow L &= \sqrt{2} \left(\frac{6\epsilon}{\pi^2} \right)^{1/3} \end{aligned} \quad (152)$$

Substituting L into eq. 150,

$$N_{S_{min}} = \frac{1 + \sqrt{1-H}}{2H}, H = 2 \times \left(1 - \frac{1}{\left(1 + \epsilon^{2/3} \left(\frac{\pi^2}{6} \right)^{1/3} \right)^2} \right) \quad (153)$$

$N_{S_{max}}$ is found by numerically solving eq. 144 with ($S_2 = 0$, $S_I = 1$, $l = 1$ and $I_k = \epsilon$) until $I_k = 1 - \epsilon$, i.e.

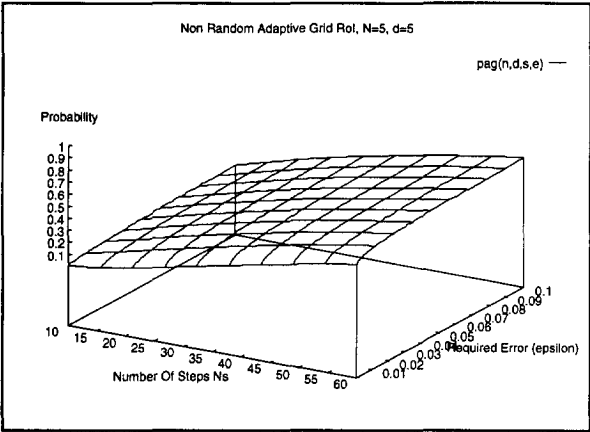


FIG. 42 The Rol for the *Non-Random Adaptive Grid* increases with number of steps and the required error.

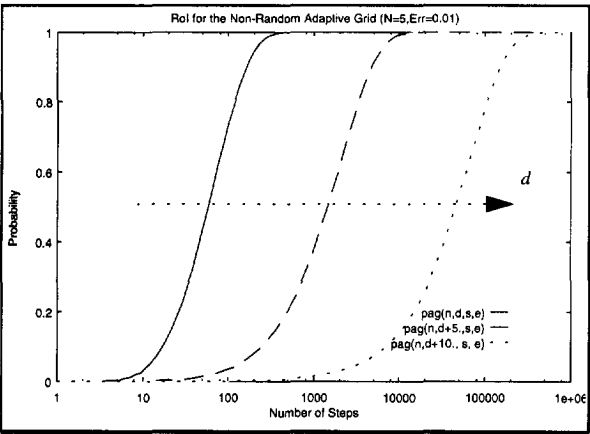


FIG. 43 The effect of dimensionality on Rol for the *Non-Random Adaptive Grid*.

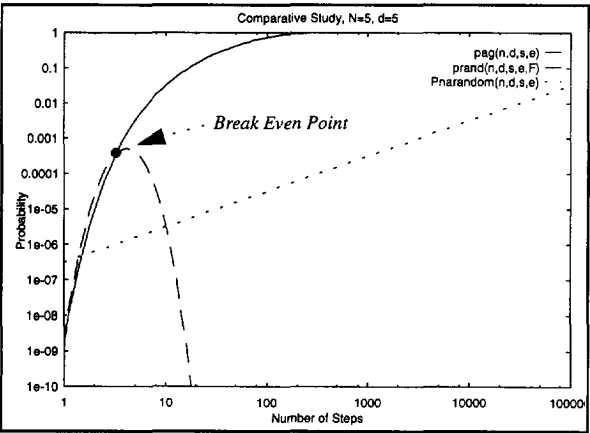


FIG. 44 Comparison of Rol between the *Non-Random Adaptive Grid* (solid) the *Adaptive Random* (dashed) and the *Simple Random* (dotted) search techniques.

FIG. 45 Effect of d and N on RoI

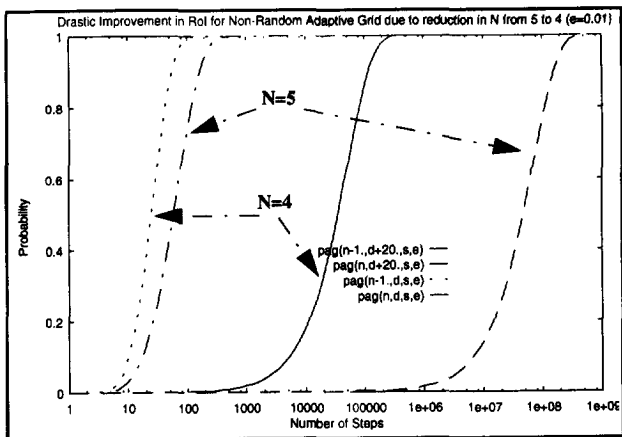


FIG. 46 Maximum(solid) and minimum (dashed) constraints on the number of steps as function of required error.

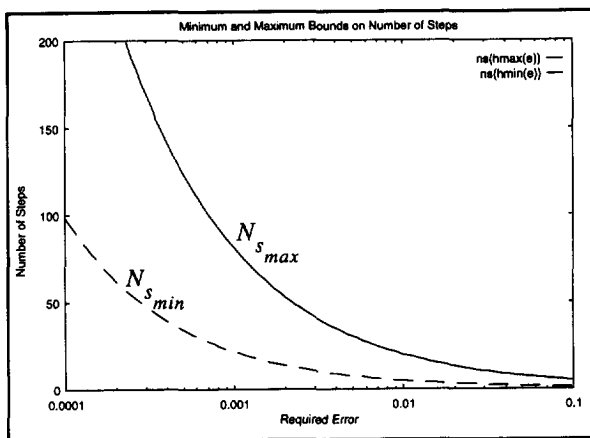
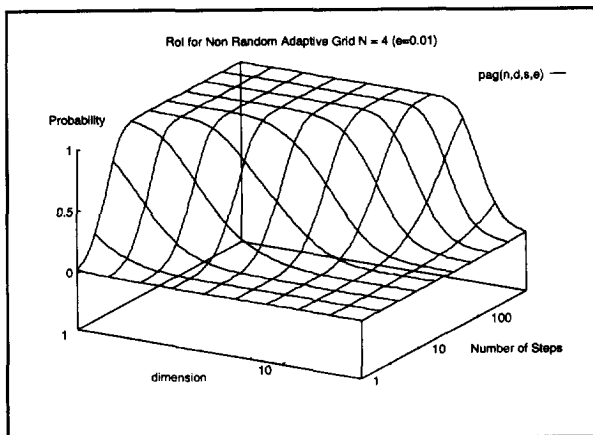


FIG. 47 RoI for the Non-Random Adaptive Grid with $N=4$ instead of five.



$$\left\{ \begin{array}{l} \delta_1 = 1 \times \frac{1-\epsilon(N-1)}{\pi N_S} \sin(\pi \epsilon) \\ \delta_2 = 2 \times \frac{1-\epsilon(N-1)}{\pi N_S} \sin(\pi(\epsilon + \delta_1)) \\ \delta_3 = 3 \times \frac{1-\epsilon(N-1)}{\pi N_S} \sin(\pi(\epsilon + \delta_1 + \delta_2)) \\ \dots \\ \delta_{N_S} = N_S \times \frac{1-\epsilon(N-1)}{\pi N_S} \sin\left(\pi\left(\epsilon + \delta_1 + \delta_2 + \dots + \delta_{N_S-1}\right)\right) \\ \left(\epsilon + \delta_1 + \delta_2 + \dots + \delta_{N_S-1} + \delta_{N_S}\right) \geq (1-\epsilon), \text{ If } (N_S \geq N_{S_{max}}) \end{array} \right\} \quad (154)$$

The above solution was numerically found for various values of ϵ and N_S . It was found to correlate very well with the following expression, which has the same form as the equation for $N_{S_{min}}$,

$$N_{S_{max}} = \frac{1 + \sqrt{1-H}}{2H}, H = 2 \times \left(1 - \frac{1}{\left(1 + \frac{\epsilon^{0.61458}}{4.6281} \right)^2} \right) \quad (155)$$

6.1.7 The Effect Of Randomness On Adaptation (*Break-Even Point*)

In the *Adaptive Random Search*, the search points are generated randomly and initially equally distributed in the solution space (S). Then a best point is chosen and the next set of search points are more likely to fall near the old best point. A new best point is then selected and the procedure continued until the concentration of search points near the best point is high enough to satisfy the accuracy requirements.

Fig. 41 visualizes the process in 2-d. Let R be hypercube of size $(1-\epsilon)^d$. Then the probability that a search point will be generated in R is also $(1-\epsilon)^d$. Search points are always generated in all of S but they are likely to fall in the region R which is reducing in size as steps go on. Finally the solution stops when 2^d points fall within the error hypercube which hopefully encloses the optimum O .

If we keep N and N_S the same, then the *Adaptive Random Search* can be considered fundamentally the same as the *Non-Random Adaptive Search* with the exception that the search points are randomly generated and it is not possible to achieve shrinkage (eq. 138) decisively. In other words, the search points are likely to follow shrinkage as desired, however this is not guaranteed and some points may fall outside of the desired Subregion. Hence the *Coupling* relations for the *Non-Random* method can be used with substitution of K_{EX} (expected shrinkage) instead of the required shrinkage K . Further if the shrinkage is different, then the final best Subregion will not have the required size ϵ^d , which needs to be taken into account.

The original size of R is $(1 - \epsilon)^d$. If the size of R is stepwise reducing by a constant factor C , then the size of R after N_S steps is $(1 - \epsilon)^d C^{N_S}$. Now it is required that m points out of M points will fall in Subregion of size ϵ^d after the final step. The probability that one point will fall in this error hypercube is the product of the probability that one point falls in R and the probability that one point falls in ϵ^d hypercube if it falls in R , i.e. $(1 - \epsilon)^d \times \epsilon^d / \left((1 - \epsilon)^d C^{N_S} \right) = \frac{\epsilon^d}{C^{N_S}}$. The expected number of points that fall in out of M points is $M \times \frac{\epsilon^d}{C^{N_S}}$ which should be m , hence

$$C^{N_S} = \frac{M \times \epsilon^d}{m} \quad (156)$$

Now there are $(N - 3)^d$ Subregions in R , so the average size of a Subregion (D) at step J is,

$$\text{Sizeof } (D) = \frac{\text{Sizeof } (R)}{(N - 3)^d} = \frac{(1 - \epsilon)^d C^J}{(N - 3)^d} \quad (157)$$

The chosen Subregion D is then shrunk by a constant factor K_{EX} to become D'' . The probability of a search point falling in D'' is,

$$\begin{aligned}
 & Prob(\text{point falls in } D'') = Prob(\text{point falls in } R) \\
 & \quad \times Prob(\text{point falls in } D \text{ if in } R) \\
 & \quad \times Prob(\text{point falls in } D'' \text{ if in } D) \\
 & = (1 - \epsilon)^d \times \frac{(1 - \epsilon)^d C^J}{(1 - \epsilon)^d C^{J+1}} \times K_{EX} = \frac{(1 - \epsilon)^d K_{EX}}{C (N - 3)^d}
 \end{aligned} \tag{158}$$

The expected number of points to fall in D'' , out of M search points should be m , i.e.

$$\begin{aligned}
 M \times Prob(\text{point falls in } D'') &= m \Rightarrow M \times \frac{(1 - \epsilon)^d K_{EX}}{C (N - 3)^d} = m \\
 \Rightarrow K_{EX} &= \frac{m C (N - 3)^d}{M (1 - \epsilon)^d}
 \end{aligned} \tag{159}$$

Substituting for C ,

$$K_{EX} = \frac{m}{M} \frac{\left(\frac{M \times \epsilon^d}{m} \right)^{\frac{1}{N_S}} (N - 3)^d}{(1 - \epsilon)^d} \tag{160}$$

The expected shrinkage is therefore different from the shrinkage required. eq. 136 & 160 then give the expected error,

$$\epsilon_{EX} = \frac{K_{EX}^{N_S/d}}{N - 1} = \left[\frac{m}{M} \frac{\left(\frac{M \times \epsilon^d}{m} \right)^{\frac{1}{N_S}} (N - 3)^d}{(1 - \epsilon)^d} \right]^{\frac{N_S}{d}} \frac{1}{N - 1} \tag{161}$$

If K is replaced by K_{EX} in eq. 138, then the probability of the *Random Adaptive Grid* is found which corners the optimum to within ϵ_{EX} and not to within ϵ . So the probability value has to be adjusted as follows,

$$\begin{aligned}
 & \text{P}_{\text{Adaptive Grid with } K_{EX} \text{ instead of } K} \searrow \\
 & \text{Prob (Optimum found within } \epsilon) = \text{Prob (Optimum found within } \epsilon_{EX}) \times \\
 & \quad \times \left(\frac{\epsilon}{\epsilon_{EX}} \right)^d \quad (162)
 \end{aligned}$$

$$\boxed{\therefore P_{\text{Random Adaptive Grid}} = \left[1 - (1 - P_{\text{Grid}} K_{EX})^{N_s} \right] \times \left(\frac{\epsilon}{\epsilon_{EX}} \right)^d} \quad (163)$$

The term $\frac{m}{M}$ can take on different values depending on how randomly the search points are generated. In general we need 2^d points to fall within the error hypercube after the final step, out of a total of $(N-2)^d$ points. If all the points were generated randomly or independently then,

$$\frac{m}{M} = \frac{2^d}{(N-2)^d} \quad (164)$$

but it is not essential to generate all the points totally independently from each other. To generate $(N-2)^d$ points in the d -dimensional solution space, it is possible to generate only $(N-2)$ points and then there is enough information to generate the rest of the search points. If only two points fall in the final error hypercube, then the rest of the points ($2^d - 2$ points) derived from these two points will also fall in. In this case,

$$\frac{m}{M} = \frac{2}{(N-2)} \quad (165)$$

As the $\frac{m}{M}$ term influences the expected shrinkage K_{EX} (eq. 159), the degree of randomness in generation of search points influences the *Coupling*.

So the fundamental difference between the random and the non-random arises due to the expected shrinkage being different to the required shrinkage. It is now possible to derive the *Break Even* point between the random and the non-random cases by equating the shrinkage factors of the two methods, and extracting the break even value of N_S ,

$$K_{EX} = K \Rightarrow \frac{m}{M} \frac{\left(\frac{M \times \epsilon^d}{m} \right)^{\frac{1}{N_S}} (N-3)^d}{(1-\epsilon)^d} = (\epsilon(N-1))^{\frac{d}{N_S}} \quad (166)$$

$$\frac{m}{M} \frac{(N-3)^d}{(1-\epsilon)^d} = \left(\frac{\epsilon^d (N-1)^d m}{\epsilon^d M} \right)^{\frac{1}{N_S}}$$

$$N_{S_{\text{Break Even}}} = \frac{\log \left((N-1)^d \frac{m}{M} \right)}{\log \left(\frac{m}{M} \frac{(N-3)^d}{(1-\epsilon)^d} \right)} \quad (167)$$

Above this break even point, the *Non-Random Adaptive Grid* is advantageous as compared with its *Random* opponent. Simulations have shown that $N_{S_{\text{Break Even}}}$ is usually very low (e.g. ~3) far less than $N_{S_{\text{min}}}$. Small values of N_S represent very low values of *Probability*. Therefore,

- If any significant measure of *Probability* is required, then the *Non-Random Adaptive Grid* gives much better *Return* (or *Probability*) on *Investment* (or *Calculation Effort*) for a given required *Accuracy*.

6.2 Variations of the Basic Theme

6.2.1 Inclusion of Constraints In the Search Strategy

Inclusion of any type of *Inequality Constraint* (e.g. linear or non-linear constraints on independent and/or dependant variables) is very simple to include since the strategy here is a *NGG search*. In the *Adaptive Grid* search, a *Best Point* is selected and then the grids are adapted towards this point. In order to include Any kind of inequality constraints, it is sufficient to state,

- The *Best Point* is the best grid point which offers the best F and which does not violate any constraints.

The above works only if there is at least one grid point that satisfies all constraints at step 0 before the optimization commences. At the start, the position of the edges and two of the grid lines is fixed (Fig. 39).

If none of the $(N-1)^d$ grid points lying on these lines satisfies the constraints, then the rest of the grid points (i.e. $N^d - (N-1)^d$ points) must be manoeuvred until at least one point is found which satisfies the constraints.

Another variation is to set up penalty relations for not satisfying the constraints, then selecting the grid point with the least penalty might eventually lead to optimum,

- If none of the grid points currently satisfy the constraints, then the current best point will be the least offender.

In practice, *Equality Constraints* can always be transformed into *Inequality Constraints* because in the physical world, there is usually a *Sense* in which the equality is meant.

6.2.2 Multiple optima

Having found the *True Optimum* O of function F , the rest of the optima (O_1, O_2, \dots etc.) are found by repeating the same procedure as above with the following addition to the rule,

- The 2^d points immediately surrounding O are not allowed to be selected as the best point, because their selection will lead to O

Once O and O_1 have been found, the procedure can be continued, but this time the points immediately surrounding O and O_1 are not allowed to be selected as best point, and so on.

If the number or the *likely* number of optima (N_{optims}) is known in advance, then the required probability can be set to the minimum value which is just sufficient to distinguish O from $O_1, O_2, \dots, O_{N_{\text{optims}}-1}$ and thereby spending no more than the necessary calculation effort. In general,

$$P_{\text{Min Required}} = \epsilon^d + \frac{N_{\text{optim}} - 1}{N_{\text{optim}}} \quad (168)$$

6.2.3 Independent Runs to Improve *RoI*

In cases where it is not important to estimate the *RoI* prior to starting the procedure, and particular knowledge exists that *F* is relatively simple (i.e. not very deceptive to search strategy), then it is possible to significantly increase *RoI* by running the *Non-Random Adaptive Grid* search with very low values of N_S , several times, independently, until at least two of the optima are the same.

If two or more independent runs yields the same optimum, then the probability that the true optimum is found is improved. This improvement can be quantified by considering two optimum points O_1 and O_2 found in independent runs,

1. $Prob(\text{point } O_1 \text{ is correct solution}) = P$
2. $Prob(\text{point } O_2 \text{ is correct solution}) = P$
3. $Prob(\text{point } O_1 \text{ is not correct solution}) = 1 - P$
4. $Prob(\text{point } O_2 \text{ is not correct solution}) = 1 - P$
5. $Prob(\text{point } O_1 \text{ is not correct solution and point } O_2 \text{ is not correct solution}) = (1 - P)(1 - P) = (1 - P)^2$

If the two points O_1 and O_2 coincide and are equivalent, then statement 5 above becomes,

6. $Prob(\text{point } O_1 \text{ is not correct solution and point } O_1 \text{ is not correct solution}) = Prob(\text{point } O_1 \text{ is not correct solution}) = (1 - P)^2$
7. $Prob(\text{point } O_1 \text{ is correct solution}) = 1 - (1 - P)^2$

Therefore, if two independent optima coalesce, then the probability that solution is correct is heavily increased. E.g. if $P=0.90$, then the combined value is $P = 1 - (1 - 0.90)^2 = 0.99$. In general if $N_{coalesce}$ runs yield the same result, then

$$P_{Coalesce} = 1 - (1 - P_{independent})^{N_{Coalesce}} \quad (169)$$

This approach requires that the starting positions of the grids be totally independent between separate runs.

6.3 Results

It is very difficult to find practical basis for comparison of various optimization techniques. If we take an example multidimensional function F and show that the *Non-Random Adaptive Grid* offers orders of magnitude better *RoI*, then it can always be argued that F was a special case, resulting in no general conclusions. Conclusive practical proof would need large number of significantly different F 's which is beyond this report.

General theoretical comparisons are also difficult because many popular optimization techniques do not provide a measure of *RoI*. The relations developed above provide basis for comparison between the *Random* and the *Non-Random Adaptive Grid*.

Fig. 42 shows the obvious that *Certainty* in the found optimum improves as more *Calculation Effort* is spent and as the required *Accuracy* diminishes.

The adverse effect of increasing dimensionality is illustrated in *Fig. 43*. Increasing the dimension of F from 5 to 15 increases the calculation effort by a factor of 100.

Fig. 44 shows that the *Non-Random Adaptive Grid* is orders of magnitude superior to the *Random Adaptive Grid* if any significant *Certainty* in the found optimum is required.

The minimum number of grid lines in each dimension is 4. (*eq. 129*). For problems with relatively low dimensions higher values of N can be used to increase confidence, without adding too much to the calculation effort. Although the relations already give an indication of how good the optimum is (i.e. a P value), we recognize that the relations are approximate and any increase in confidence is appreciated. *Fig. 45 & Fig. 47* show the effect of using different number of grid lines.

The allowable range of values for N_S reduces as the required accuracy reduces (*Fig. 46*). The upper bound on N_S (*eq. 155*) is not a serious constraint and can be violated, as long as the starting position of the grid lines are adjusted so that the grid lines do not become fully concentrated before the final steps are reached. For example, suppose $N_S=50$ is necessary with $\epsilon = 0.01$. *Fig. 46* shows that this N_S value is too high. However $N_S=50$ is allowed if the starting position of the AB and CD grid lines (*Fig. 39*) is 0.002 from the edges instead of 0.01.

6.4 Clarification of Logic Behind Derivations

6.4.1 Def2 and the Probability of Step Success

Def2 forms the basis of the relations derived. At first sight it may falsely appear that it is a “circular” definition or that success of each step depends on the success of previous step. If step successes are interdependent then they cannot be considered as Bernoulli trials making the derivations false.

The key here is the definition of step success. A step is successful if selected best point falls in a SR. If selected best point does fall in a SR, there is no guarantee/ implication that the best point chosen during the next step will also fall in a SR. The only implication or guarantee is that solution will be found in the final step. SR or SR's could be of any size or shape and could be anywhere in the solution space and can vary between the steps.

So the EVENT considered here is the EVENT of selected best point falling in a SR. The probability of step success is the probability of this event. These successive events are in general totally independent of each other. The event of selected best point falling in a SR gets no influence from the grid adaptation mechanism that relates successive steps. Of course grid adaptation in one step influences the position of the grids in the following step, but this does not imply anything about grid points falling in SR in this step that follows.

If the probability of *step success* is 100%, then we have a special case. Here the probability of success of the following steps become insignificant, because final *overall success* is guaranteed, i.e. if we are sure that a selected best point falls in a SR, it is not important any more whether or not the following points will also fall in SR during the next step. However, as the result for the probability of step success shows, a value of 100% is not achievable.

Unfortunately it is impossible to quantify the probability of step success as defined by *Def2*. Here then the conditions in *Def1* is used to estimate the value of step success in *Def2*. So the logic behind the derivations is *Def2*. The consequence of using *Def1* to estimate step success is to get a conservative estimate of overall *P*. In other words the probability of overall success of the method is actually better than that given by the derived *Coupling* relations.

6.4.2 Behaviour of F and Constraints

It is natural to think that behaviour of constraints or functions will affect the probability of overall success of a search method. This is usually true for iterative and/or gradient based methods which can either totally fooled by the behaviour of function and/or constraints or will require a longer time (i.e. larger number of steps) before they home in onto the solution. In the adaptive grid methods, however, this is the wrong way to think because, number of steps is fixed in advance and the search method could not care less how badly the function and constraints behave, as long as the function and constraints can be evaluated.

6.4.3 Rigorous Accounting of the Type of Search Points and the Number of of Affected Subregions

In deriving the gridding term in the step success equation, a distinction was made between *Vertex*, *Edge* and *Body* points. All points other than these three, were then designated *Face* points.(eq. 127). In general it is possible to distinguish d different types of points (for a d -dimensional solution space), where each different type can be assigned as order, e.g.

1. *Vertex* point has order 0
2. *Edge* point has order 1
3. *Face* point has order $d-1$
4. *Body* point has order d

The derivation of the gridding term P_{grid} above, considers all the points between order 2 up to and including order $d-1$. If we do consider all the d different types of points, the number of points of type m and their affected subregions is given by,

$$N_{points_m} = \frac{d!}{m!(d-m)!} 2^{d-m} (n-2)^m \quad (170)$$

$$N_{subregions_m} = 2^m$$

This changes the result for P_{grid} (from eq. 127) to,

$$\begin{aligned}
 P_{grid} &= \frac{\sum_{m=1}^d \frac{d!}{m! (d-m)!} 2^{d-m} (n-2)^m 2^m}{\left(\sum_{m=1}^d \frac{d!}{m! (d-m)!} 2^{d-m} (n-2)^m \right) (N-1)^d} \\
 &= \frac{\sum_{m=1}^d \frac{d!}{m! (d-m)!} 2^d (n-2)^m}{\left(\sum_{m=1}^d \frac{d!}{m! (d-m)!} 2^{d-m} (n-2)^m \right) (N-1)^d}
 \end{aligned} \tag{171}$$

eq. 174 gives smaller values for P_{grid} than eq. 127.

The reader should remember that the logic behind the derivation of the *coupling*, rests on *Def2*. But it was impossible to test for the conditions of *Def2* and hence *Def1* was used to estimate *Def2*. The consequence of this was that P_{step} is underestimated.

This is the reason why the less rigorous eq. 127 is preferred to eq. 174 because it offsets the underestimation of P_{step} .

6.5 Summary for the Non-Random Adaptive Grid Search

Return on Investment (RoI) (i.e. the *Coupling between Probability, Accuracy and Calculation Effort*) is a big issue in optimization in general. In developing modern, high speed *Design, Analysis or Modelling* systems where all the various tasks are carefully matched and tuned to give a high speed overall system, it is extremely important to have a quantitative measure of this coupling *Prior* to starting the solution procedure, if optimization is an “*In The Loop*” event.

It is possible to analytically derive the full *RoI* relations for the *Non-Random Adaptive Grid* search and further modify them to represent *Random Adaptive Grid* with various degrees of randomness. The following conclusions can be drawn from this theoretical study,

Based on the relations derived in this report, the following conclusions are made,

1. *Reduction of the Search Volume* (e.g. in the *Simplex Approach*) increasingly penalizes step success as solution proceeds and therefore heavily reduces the overall *Rol*.
2. Any degree of *Randomness* in the generation of search points causes the *Shrinkage* of the best region to occur less decisively. This translates into very poor *Rol*.
3. *Grid Adaptation* should be relatively *Slow* during the *Early* and *Final* steps of the search. It can then be relatively *Faster* in the *Intermediate* stages to make up for lost time
4. *Deflection* properties of an imaginary *Flexible Beam* can be used to adapt grid position complying with previous step.
5. The analytical relations for the *Break-Even* number of steps suggest, that if any significant level of *Probability* is required, then the *Non-Random Adaptive Grid* gives orders of magnitude better *Rol* than the *Random Adaptive Grid*.

As dimensions increase, an *explosion* occurs in the calculation effort of the *Adaptive Grid* (eq. 139). For $N=4$, $d=25$, $t=1$ ms, the first step would take 18.64 hours and every following step 9.32 hours. The minimum number of steps is 1. The required number of steps is given by required values of ϵ , P , d , N and *Coupling* eq. 138. If $d=50$ then just the first step would require 7140.4 years!. The suitability of method is for low to middle dimensional (< 25) and high speed optimization functions (\sim ms).

Calculation Explosion at high dimensions is not only restricted to this method and majority of optimization methods suffer heavily from this. The good thing about the *Non-Random Adaptive Grid* method described here is that it is honest about *Rol* and it is possible, with the aid of the coupling equation, to estimate the computation burden for a given set of F , s of ϵ , P , d and N **BEFORE** starting.

One way to avoid the *Calculation Explosion* (to a great extent, i.e d up to 100 or more) is to *Move Through the Solution Space* in *Real Time* for optimization, made possible by the high speed of F . Here the human is used in the optimization loop, and his understanding of model behaviour is used to set the direction of search. See section 2.2 for description of *Dyn_Carp* (Dynamic Carpets) and *Dyn_Hist* (Dynamic Histograms).

6.6 The Modified Multivariate Newton Raphson

The “*Multivariate Newton Raphson*” is very commonly used in engine analysis programs for function generation from multidimensional data (section 6.6.1 on page 153) and linear iteration (section 6.6.2 on page 158) because it is relatively simple, flexible,

stable and general. Provided the system of equations are close to linear and/or if the first guess is close to the final point, iterations will converge quickly.

The method simply makes a linear relation between two vectors \vec{E} and \vec{V} through a gradient matrix,

$$\vec{E} = \vec{M} \cdot \vec{V} + \vec{C} \quad (172)$$

Where,

$$\vec{E} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}, \vec{V} = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_m \end{bmatrix}, \vec{M} = \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1m} \\ M_{21} & M_{22} & \dots & M_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \dots & M_{nm} \end{bmatrix} \quad (173)$$

where,

$$M_{kj} = \frac{dE_k}{dV_j}$$

6.6.1 Multidimensional Function Generation From Data

In function generation from multidimensional data, the multidimensional function can be represented as follows,

$$E[k] [i_1] [i_2] \dots [i_m] \quad (174)$$

and the independent variable data can be represented as,

$$V[j] [i_j] \quad (175)$$

where m is number of independent variables, n is the number of dependant functions, $j = \text{Index of } V$, $i_j = \text{Index of } V_j = 1, 2, \dots \text{order of } V_j$ and $= \text{Index of } E = 1, 2, \dots$

Now the object of multidimensional function generation is to find the new \vec{E} Corresponding to new \vec{V} given sufficient data set $E[k] [i_1] [i_2] \dots [i_m]$ and $V[j] [i_j]$. The basic procedures involved are,

1. Find the multidimensional grid of order M in the data set which encloses the required data point. Stated mathematically, for each $j = 1, 2, \dots, m$ find the index i_j such that the j^{th} component of \vec{V}_{new} lies between $V[j] [i_j]$ and $V[j] [i_j + 1]$. Let the base point of this grid be represented by, \vec{V}_{ref} and \vec{E}_{ref} .
2. Calculate the gradient matrix M using the i_j 's found in step 1, as shown below,

$$M_{kj} = \frac{dE_k}{dV_j} \approx \frac{E[k] [i_1] \dots [i_j + 1] \dots [i_m] - E[k] [i_1] \dots [i_j] \dots [i_m]}{V[j] [i_j + 1] - V[j] [i_j]}$$

for
 $j = 1, 2, \dots, m$
 and
 $k = 1, 2, \dots, n$

(176)

3. Having calculated the gradient matrix M , then we find \vec{E}_{new} as follows,

$$\vec{E}_{new} = \vec{E}_{ref} + \vec{M} \cdot (\vec{V}_{new} - \vec{V}_{ref})$$

or

$$\begin{bmatrix} E_{1_{new}} \\ E_{2_{new}} \\ \vdots \\ E_{n_{new}} \end{bmatrix} = \begin{bmatrix} E_{1_{ref}} \\ E_{2_{ref}} \\ \vdots \\ E_{n_{ref}} \end{bmatrix} + \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1m} \\ M_{21} & M_{22} & \dots & M_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \dots & M_{nm} \end{bmatrix} \begin{bmatrix} V_{1_{new}} - V_{1_{ref}} \\ V_{2_{new}} - V_{2_{ref}} \\ \vdots \\ V_{m_{new}} - V_{m_{ref}} \end{bmatrix}$$
(177)

The *Multivariate Newton Raphson* as stated above is not suitable for high speed function generation if the independent and dependant dimensions of the data are high because,

1. A lot of time is wasted locating the smallest multidimensional grid enclosing \vec{V}_{new} . Here each component of \vec{V}_{new} must be compared with every single corresponding value in the data set. For example if there are ten values of V_2 in the data set, then $V_{2_{new}}$ must be compared with each of these values to determine where the new point sits.

2. Once the new point is located, many division and subtractions are required in eq. 176 to determine the elements of the gradient matrix.

However with a couple of very simple tricks, it is possible to eliminate the former procedure and heavily accelerate the latter. The trick is simply,

- Transform the problem such that the integer indices are considered as the real independent variables and \vec{V} is considered as the dependant variable by making simple reversible relationsⁱ between the indices and the component of the \vec{V} vector.

We now need m simple reversible relations between V_j and index of V_j which is i_j in eq. 175,

$$\begin{aligned}
 V_1 &= f_1(i_1) & i_1 &= \text{arcf}_1(V_1) \\
 V_2 &= f_2(i_2) & i_2 &= \text{arcf}_2(V_2) \\
 &\vdots & & \\
 V_m &= f_m(i_m) & i_m &= \text{arcf}_m(V_m)
 \end{aligned} \tag{178}$$

or

$$\vec{V} = F_{\text{reversible}}(\vec{I})$$

- Transform the n -dimensional function E of eq. 174 into n separate ΔE_i functions, where each ΔE_i function gives the change in function values in the i^{th} direction at every grid point, as shown below,

$$\begin{aligned}
 \Delta E_1[k] [i_1] [i_2] \dots [i_m] \\
 \Delta E_2[k] [i_1] [i_2] \dots [i_m] \\
 \vdots \\
 \Delta E_m[k] [i_1] [i_2] \dots [i_m]
 \end{aligned} \tag{179}$$

-
- i. These can be simple linear, logarithmic or any relation that is quick to evaluate and is reversible. e.g. $\{ V_1 = 5.0 \times i_1 + 3, \quad i_1 = (V_1 - 3.0) / 5.0 \}$

where $\Delta E_i[k]$ gives the k^{th} function increment along the i^{th} direction, at a given grid point, i.e. at fixed values of i_1, i_2, \dots, i_m ,

$$\begin{aligned} \Delta E_i[k] [i_1] [i_2] \dots [i_m] = \\ E[k] [i_1] [i_2] \dots [i_i + 1] \dots [i_m] - E[k] [i_1] [i_2] \dots [i_i] \dots [i_m] \end{aligned} \quad (180)$$

Then at any multidimensional grid point defined by fixed values of i_1, i_2, \dots, i_m the function E is evaluated as follows,

$$\begin{aligned} E[k] [i_1] [i_2] \dots [i_m] = E[k] [1] [1] \dots [1] + \sum_{l=1}^{i_1} \Delta E_1[k] [l] [i_2] \dots [i_m] + \\ + \sum_{l=1}^{i_2} \Delta E_2[k] [i_1] [l] \dots [i_m] + \dots + \sum_{l=1}^{i_m} \Delta E_n[k] [i_1] [i_2] \dots [l] \end{aligned} \quad (181)$$

Now eq. 172 is transformed into,

$$\boxed{\vec{\Delta E} = \vec{M} \cdot \vec{I}} \quad (182)$$

and eq. 176 is transformed into,

$$f_{kj} = \left(\frac{dE_k}{dI_j} \approx \frac{\Delta E_j[k] [i_1] [i_2] \dots [i_m]}{\underbrace{I[j] [(int) i_j + 1] - I[j] [(int) i_j]}_{\text{Unity}}} \right) = \Delta E_j[k] [i_1] [i_2] \dots [i_m] \quad (183)$$

$$\begin{aligned} & j = 1, 2, \dots, m \\ & \text{and} \\ & k = 1, 2, \dots, n \end{aligned}$$

The (int) operator above returns the integer part of the real number i_j , e.g. $(int)1.5897 = 1$.

The steps for multidimensional function generation are now modified, starting with given data set and a new independent variable vector \vec{V}_{new} ,

1. Process the function data and split the E function table into ΔE_i such that eq. 181 holds and store ΔE_i tables.
2. Use the $arctf_i(V_i)$ functions of eq. 178 to calculate the \vec{I}_{new} vector corresponding to \vec{V}_{new} . This implies that the base point or reference point of the smallest multidimensional grid enclosing the \vec{V}_{new} is directly determined by,

$$I_{ref}[j] = (int) I_{new}[j] \quad \text{for } j = 1, 2, \dots, m \quad (184)$$

3. Calculate $\vec{\Delta I}_{ref}$ by extracting the fractional parts of \vec{I}_{new} ,

$$\Delta I_{new}[j] = (fract) I_{new}[j] \quad \text{for } j = 1, 2, \dots, m \quad (185)$$

where $(fract)X$ returns the fractional part of the floating point number X , e.g.
 $(fract) 1.5897 = 0.5897$.

4. Calculate \vec{E}_{ref} corresponding to \vec{I}_{ref} from eq. 181. Also extract $\vec{\Delta E}_{ref}$
5. Calculate the M matrix from eq. 183 at this \vec{I}_{ref}
6. then find \vec{E}_{new} from,

$$\begin{aligned} \vec{E}_{new} &= \vec{E}_{ref} + \vec{M} \cdot \vec{\Delta I}_{new} \\ \text{or} \\ \begin{bmatrix} E_{1_{new}} \\ E_{2_{new}} \\ \vdots \\ E_{n_{new}} \end{bmatrix} &= \begin{bmatrix} E_{1_{ref}} \\ E_{2_{ref}} \\ \vdots \\ E_{n_{ref}} \end{bmatrix} + \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1m} \\ M_{21} & M_{22} & \dots & M_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \dots & M_{nm} \end{bmatrix} \begin{bmatrix} \Delta I_{new_1} \\ \Delta I_{new_2} \\ \vdots \\ \Delta I_{new_m} \end{bmatrix} \end{aligned} \quad (186)$$

7. Repeat steps 2 to 6 above for any new value state of \vec{V}_{new} vector. Step 1 effort remains valid as long as the original data are not changed.

It should now be clear to the reader that the above modification has eliminated totally all value comparisons for locating the reference point of the multidimensional grid. Also all subtractions and divisions in calculating the gradient matrix M have been discarded, resulting in very high speed method for function generation at no cost to accuracy of method.

6.6.2 Linear Solution to n Simultaneous Equations

In Off-Design Cycle Analysis, *Multivariate Newton Raphson* is very often used to find solution to n equations and n unknown type of problems because of its simplicity and generality. Here the \vec{E} vector is set to represent the error vector and the equations are set up such that each component of \vec{E} is given in terms of the \vec{V} vector. In this case the n and m indices are equal i.e. the M matrix in eq. 173 is a square matrix.

The starting point is a given relation between \vec{E} and \vec{V} ,

$$\begin{aligned} \vec{E} &= \text{func}(\vec{V}) \\ \text{or} \\ \left\{ \begin{array}{l} E_1 = \text{func}_1(V_1, V_2, \dots, V_n) \\ \vdots \\ E_n = \text{func}_n(V_1, V_2, \dots, V_n) \end{array} \right\} \end{aligned} \quad (187)$$

1. Find a good first guess or reference condition \vec{V}_{ref} and find corresponding \vec{E}_{ref} from eq. 187.
2. Set the \vec{E}_{new} vector to null.
3. Calculate the elements of gradient matrix M from eq. 188 corresponding to \vec{V}_{ref} , by perturbing each independent variable in turn as shown below,

$$M_{ij} = \frac{dE_i}{dV_j} = \frac{1}{\delta V_j} \left[\text{func}_i(V_{1_{ref}}, V_{2_{ref}}, \dots, V_{j_{ref}} + \delta V_j, \dots, V_{m_{ref}}) - \text{func}_i(V_{1_{ref}}, V_{2_{ref}}, \dots, V_{j_{ref}}, \dots, V_{m_{ref}}) \right] \quad (188)$$

4. Calculate the inverse of matrix M .
5. Find \vec{V}_{new} corresponding to null \vec{E}_{new} from the following,

$$\vec{E}_{new} - \vec{E}_{ref} = \vec{M} \cdot (\vec{V}_{new} - \vec{V}_{ref}) \quad \vec{E}_{new} = [0, 0, \dots, 0] = null \quad (189)$$

multiplying both sides by inverse of matrix M gives,

$$\begin{aligned} \vec{V}_{new} &= \vec{V}_{ref} - (Inv) \vec{M} \cdot \vec{E}_{ref} \\ \text{or} \\ \begin{bmatrix} V_{1_{new}} \\ V_{2_{new}} \\ \vdots \\ V_{n_{new}} \end{bmatrix} &= \begin{bmatrix} V_{1_{ref}} \\ V_{2_{ref}} \\ \vdots \\ V_{n_{ref}} \end{bmatrix} - \begin{bmatrix} M_{11} & M_{12} & \dots & M_{1n} \\ M_{21} & M_{22} & \dots & M_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \dots & M_{nn} \end{bmatrix}^{-1} \begin{bmatrix} E_{1_{ref}} \\ E_{2_{ref}} \\ \vdots \\ E_{n_{ref}} \end{bmatrix} \end{aligned} \quad (190)$$

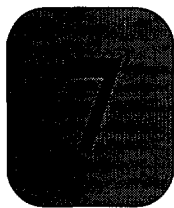
6. Calculate \vec{E}_{new} from eq. 187 using \vec{V}_{new} from step 5.
7. Let \vec{V}_{ref} be equal to \vec{V}_{new} and repeat steps 1 to 5 until all the components of \vec{E}_{new} are acceptably close to zero.

The above method has the following limitations,

- Either the original set of functions in eq. 188 must not be too non-linear and/or the first “good” guess must be good.
- The number of independent variables must be equal to the number of error variables. Also the gradient matrix M must behave properly such that its inverse can always be found.

The speed and accuracy of the above procedure can be improved by improving the first guess. This is generally problem dependant. For Engine Thermodynamic Off-Design Analysis it is possible to analytically derive the solution to an accuracy of around 5%. See section 4.4 .

Also the problem can be rewritten in terms of the index vector instead of the independent variable vector, as in the previous section, in order to eliminate divisions in evaluation of matrix M in eq. 188.



Validation of Methods and Progress

The majority of topics and methods discussed in this thesis can be classified as non-standard work and it was felt necessary to include a chapter on “*validation*”. The term “*validation*” can have a broad meaning in general. Full validation of the proposed methodology for conceptual/preliminary design of aircraft engines would in the first place require that the *CAGED* system is fully implemented and mature. It would then be necessary to show that *CAGED* can find particular design solutions which would be disadvantageous or impossible to find using other conceptual/preliminary design systems. This kind of validation is a huge task and beyond the scope of this report. However it is possible to isolate some typical preliminary/conceptual design activity or tasks and show that using *CAGED* is much more advantageous than other programs. To the best of the Author's knowledge, a conceptual/preliminary design system which employs the *Natural Design Cycle* philosophy does not exist elsewhere to aid any comparison. All other systems known to the author can be classified as *Analysis* programs and not *Design* programs. Using an analysis program, it is possible to carry out a design task by making many point analysis. In section 8.3.5, some simple tasks are performed with *CAGED* and *GASTURB*^[36] (a good example of an analysis program) and a rough comparison of computational burden (i.e. execution times) is made.

This chapter starts with validating the *on-design* thermodynamic engine models for nine example unmixed turbofan engines by comparison with *GASTURB* output. The off-design thermodynamic relations derived previously have not yet been incorporated into the generated engine models due to lack of time. However the off-design relations themselves are validated by comparison with *GASTURB* output for an unmixed turbofan in section 7.2. As discussed previously, the structure of off-design engine

models generated by *CAGED* are almost identical to the on-design engine models with the exception of updating some generic spool variables. Hence the validation effort here is sufficient to show the quality of the off-design engine models in *CAGED* when fully implemented.

The preceding chapter derived a new numerical optimization technique suitable for difficult multi-dimensional and high speed real time merit functions. In section 7.3, this technique is successfully tested on a particularly difficult (multiple optima and discontinuous) 3-dimensional function which would confuse and mislead the majority of gradient guided search techniques.

A major proportion of this research involved developing the *GUI*, *DynCarp* and *DynHist* programs. These are new programs and had to be purpose written due to stringent performance requirements. It would be very easy to convince the reader that the programs really work and that they are powerful tools in design work, if he/she could try out the *CAGED* system on a workstation.

7.1 Validation of on-design engine models

An unmixed turbofan configuration was defined in *CAGED* whose major cycle parameters are listed in Table 2. The unmixed turbofan cycle option of *GASTURB* was then used to provide specific fuel consumption and specific thrust data for the parameter space spanned by the above table, a total of 225 design points for the fixed engine configuration.

The resulting output from the two programs are plotted in *Fig. 48* up to and including *Fig. 56*. Each of these figures are in quadruple plot format with the following key,

1. The top left plot is a surface plot of specific thrust versus bypass ratio and turbine entry temperature. At the base of surface, contours denote constant specific thrust values.
2. The top right plot is a surface plot of specific fuel consumption versus bypass ratio and turbine entry temperature. At the base of surface, contours denote constant specific fuel consumption values.
3. Bottom left plot compares specific thrust values from *GASTURB* (x value of data points) and *CAGED* (y value of data points). The lines indicate the range $\pm 1\%$.
4. Bottom right plot compares specific fuel consumption values from *GASTURB* (x value of data points) and *CAGED* (y value of data points). The lines indicate the range $\pm 1\%$.

TABLE 2

Major cycle Parameters of example two spool unmixed turbofan

<i>Parameter</i>	<i>Values</i>
intake polytropic efficiency	1.00
fan pressure ratio	1.8, 2.5, 3.0
overall pressure ratio	18, 25, 50
fan polytropic efficiency	0.9079
compressor polytropic efficiency	0.9044
burner pressure ratio	0.97
burner combustion efficiency	0.99
bypass ratio	1.0, 2.0, 3.0, 4.0, 5.0
turbine entry temperature	1400, 1500, 1600, 1700, 1800 °K
hp-turbine polytropic efficiency	0.91
lp-turbine polytropic efficiency	0.92
core nozzle polytropic efficiency	1.00
bypass nozzle polytropic efficiency	1.00
inter-duct pressure ratios	1.00
bleeds	0.0 both spools
mechanical efficiency	1.00 both spools
shaft power off-take	0.0 both spools
altitude	0 m, 11000 m
flight mach number	0, 0.4, 0.8

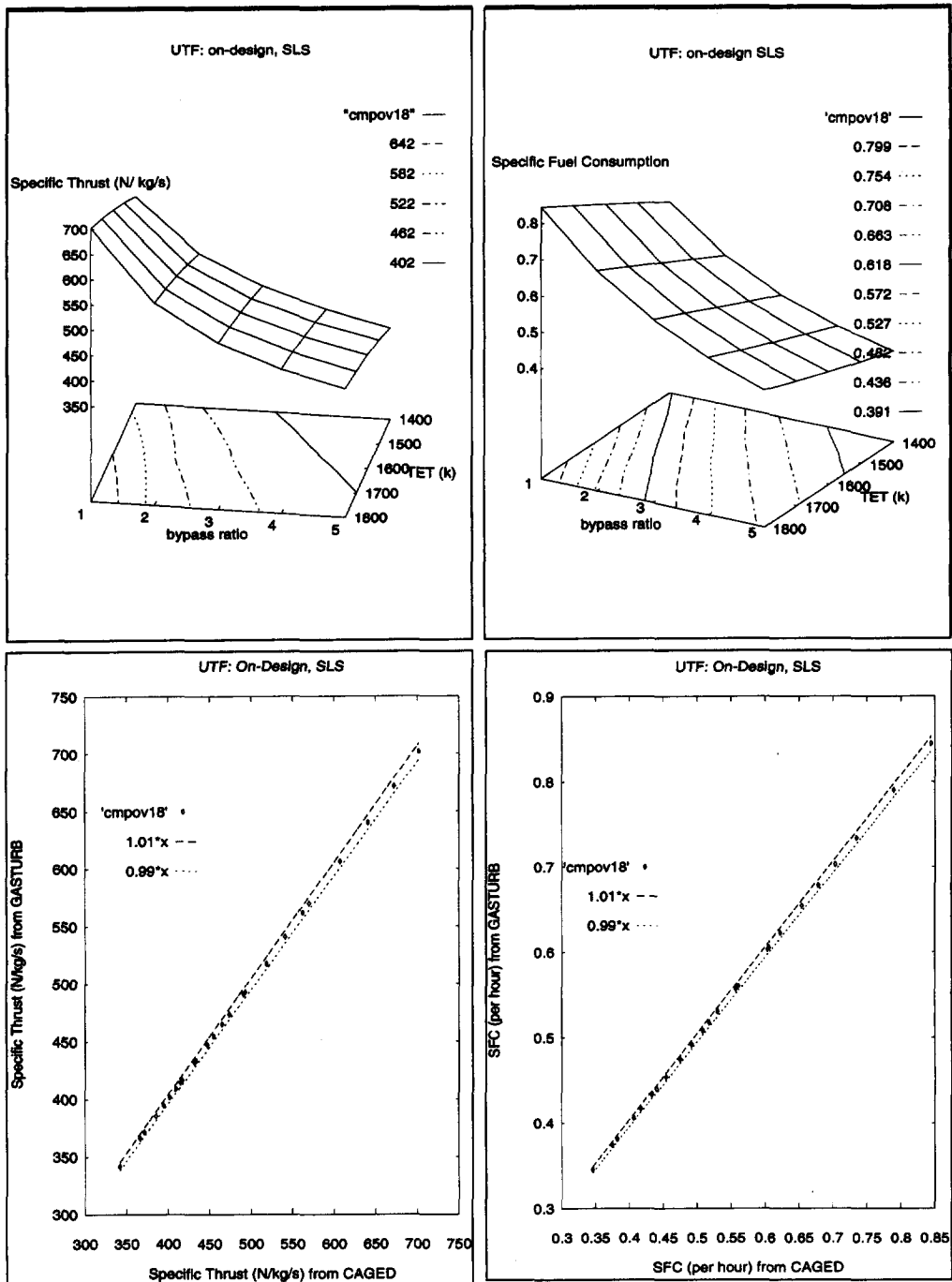


FIG. 48 On-design performance of unmixed turbofan at SLS conditions. Major cycle parameters: overall pressure ratio=18, fan pressure ratio=1.8, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

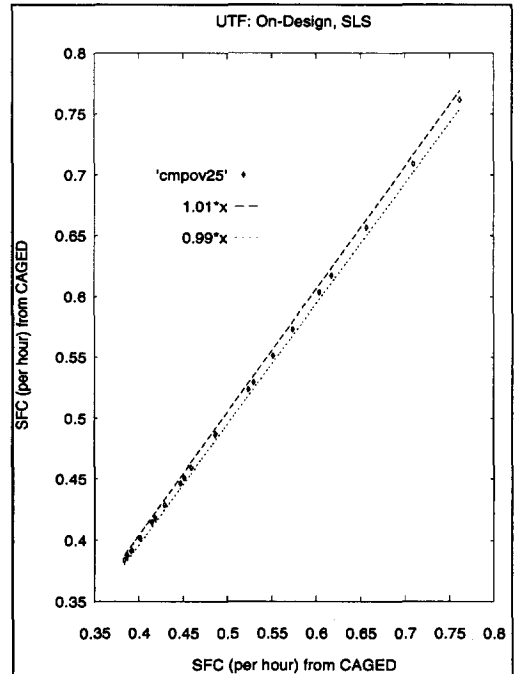
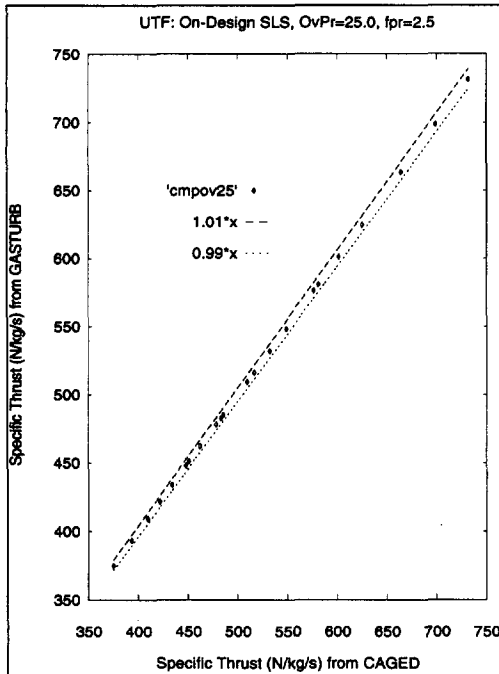
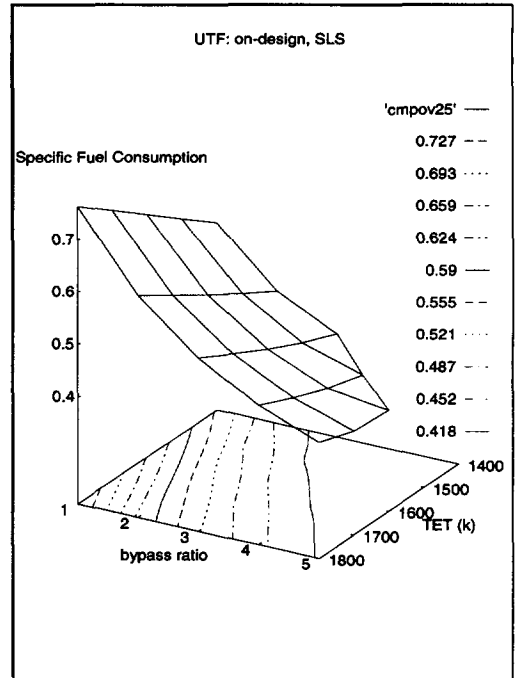
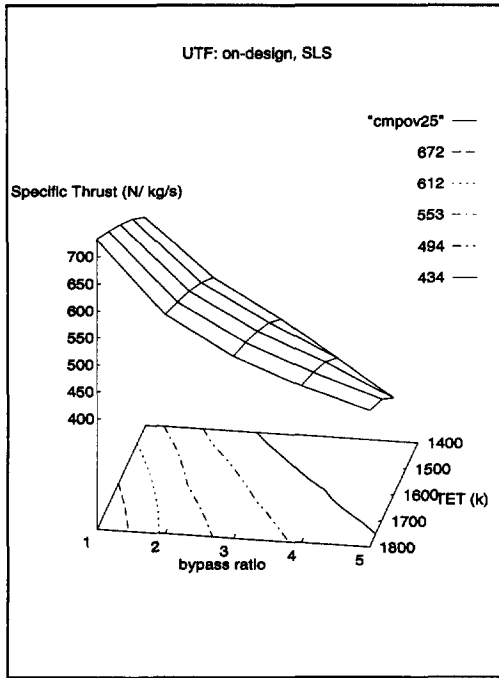


FIG. 49 On-design performance of unmixed turbofan at SLS conditions. Major cycle parameters: overall pressure ratio=25, fan pressure ratio=2.5, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

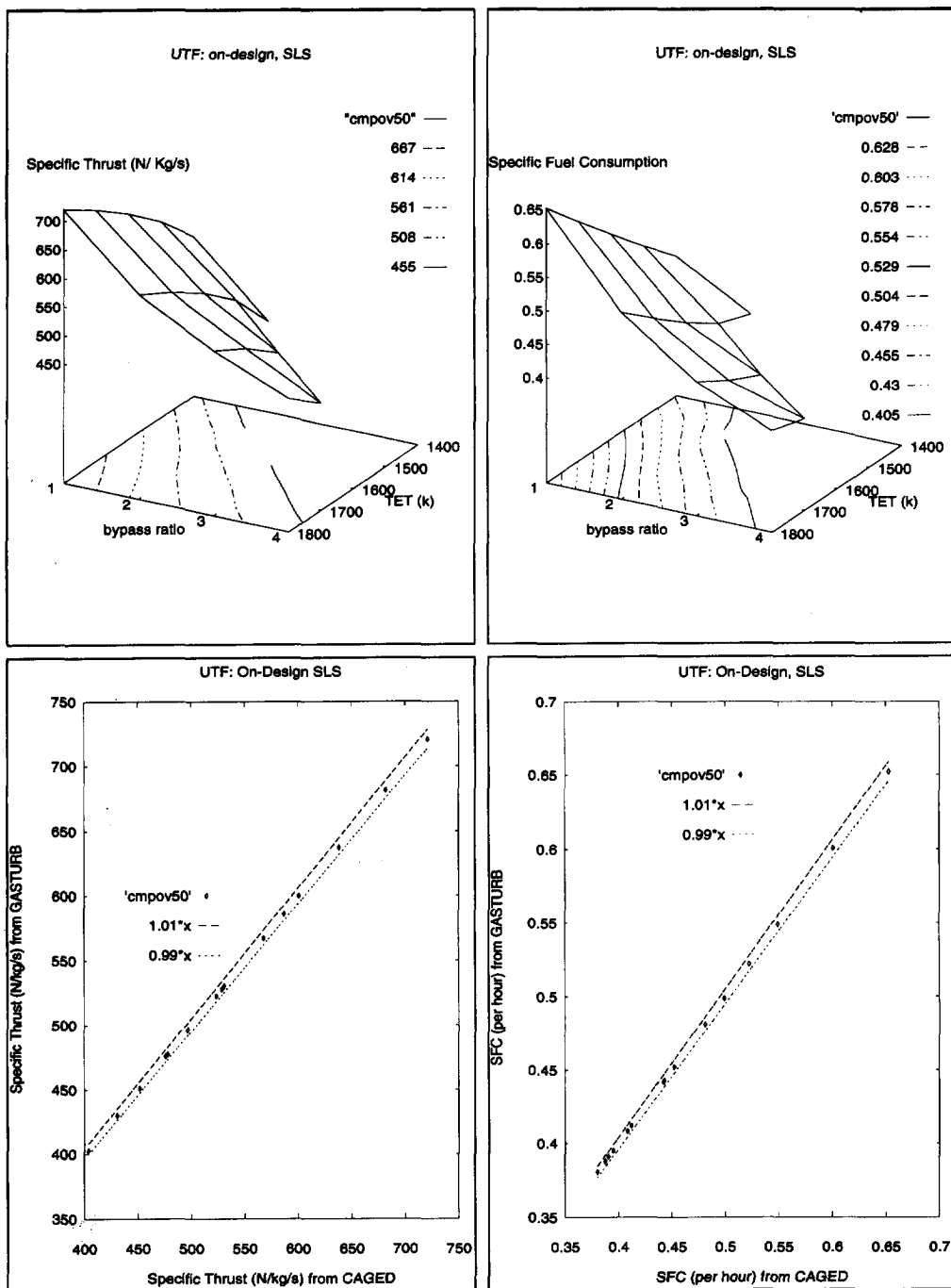


FIG. 50 On-design performance of unmixed turbofan at SLS conditions. Major cycle parameters: overall pressure ratio=50, fan pressure ratio=3.0, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

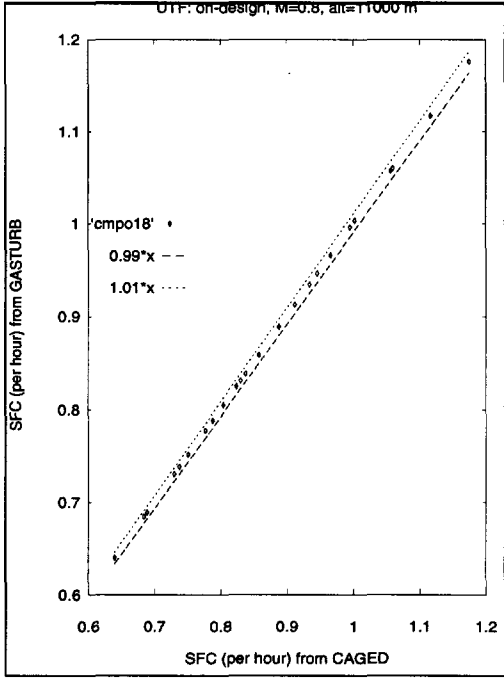
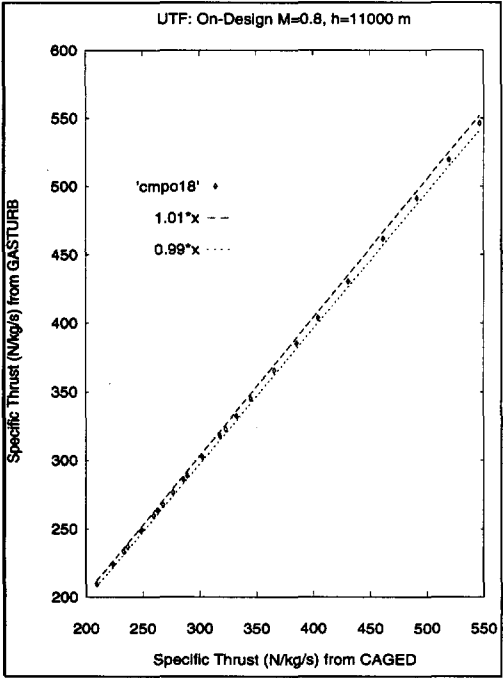
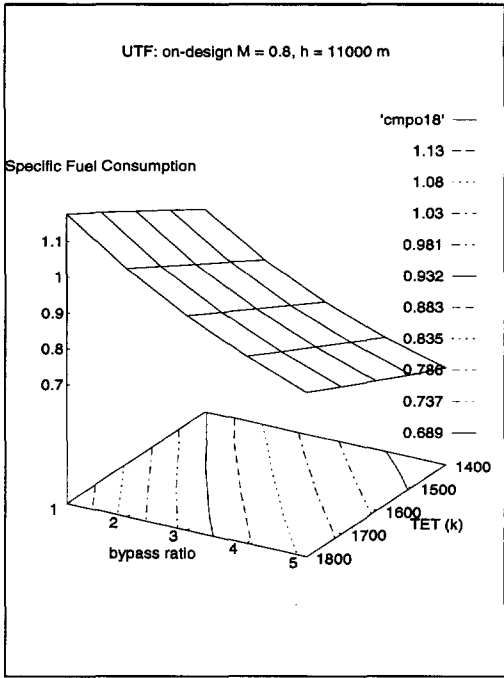
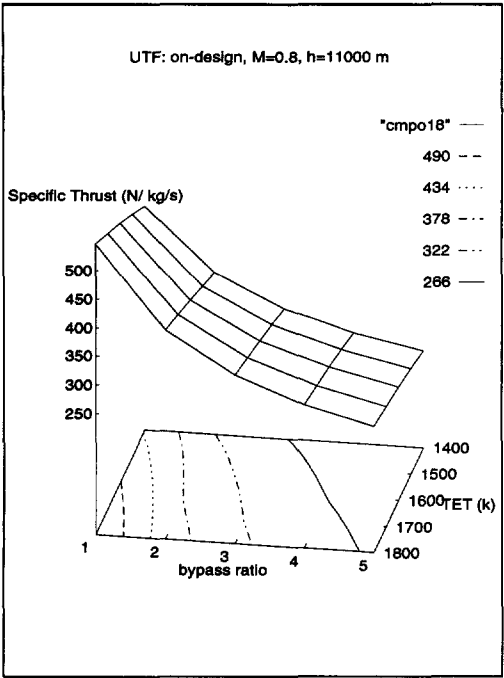


FIG. 51 On-design performance of unmixed turbofan at $M = 0.8$, $h = 11000$ conditions. Major cycle parameters: overall pressure ratio = 18, fan pressure ratio = 1.8, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

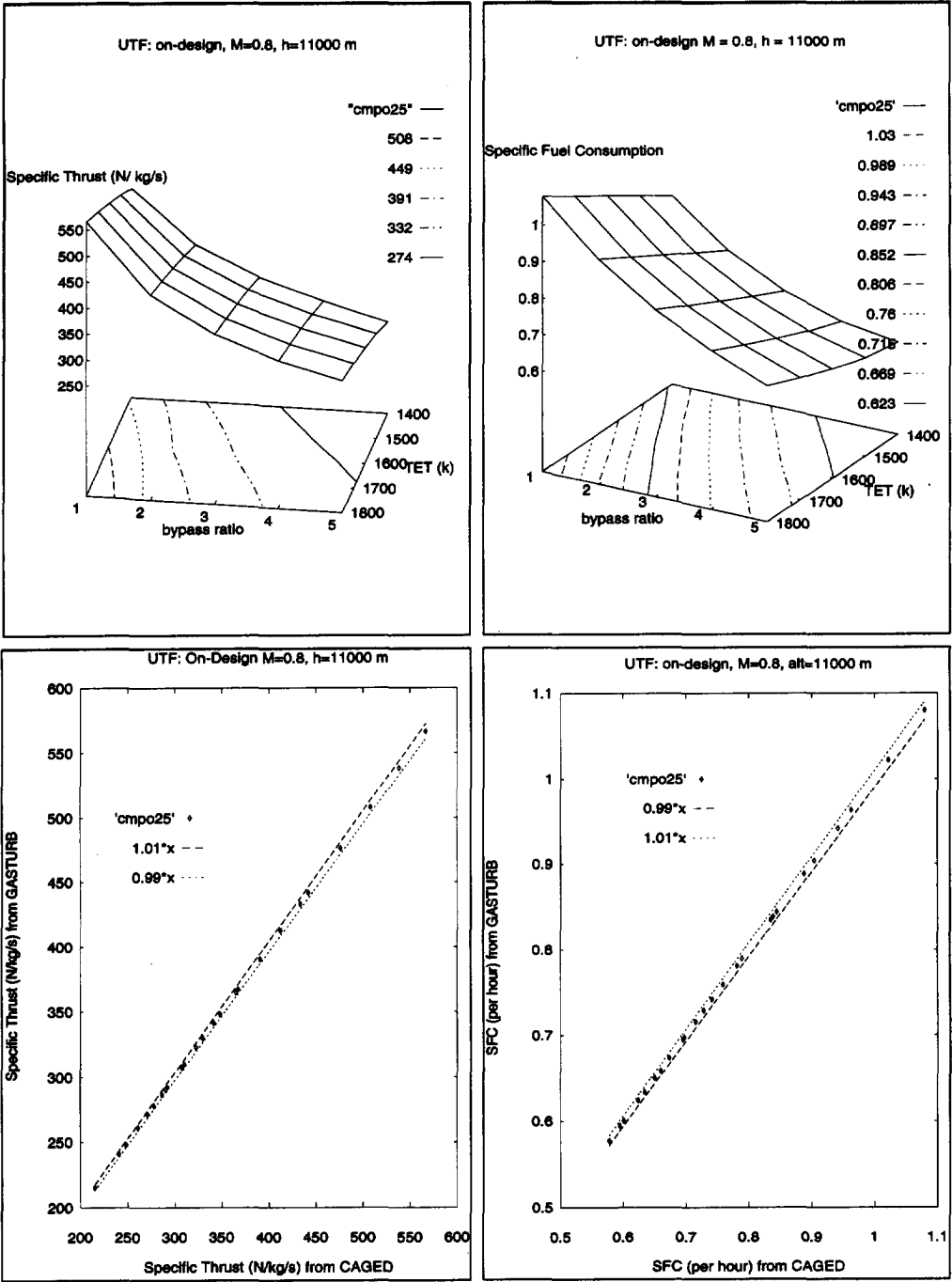


FIG. 52 On-design performance of unmixed turbofan at $M = 0.8$, $h = 11000$ conditions. Major cycle parameters: overall pressure ratio = 25, fan pressure ratio = 2.5, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

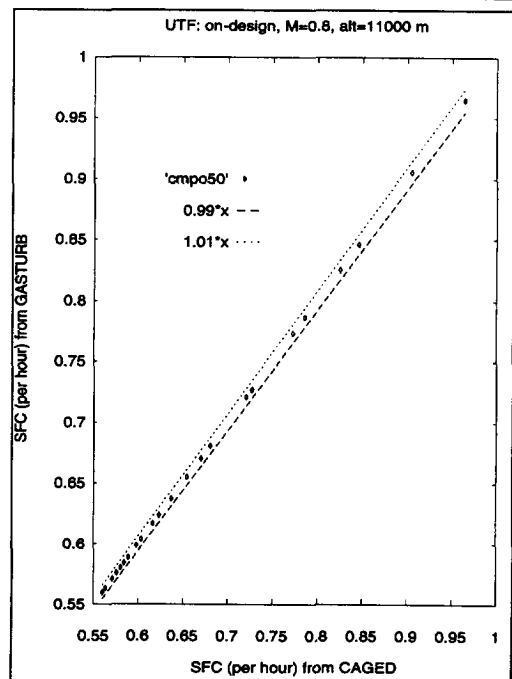
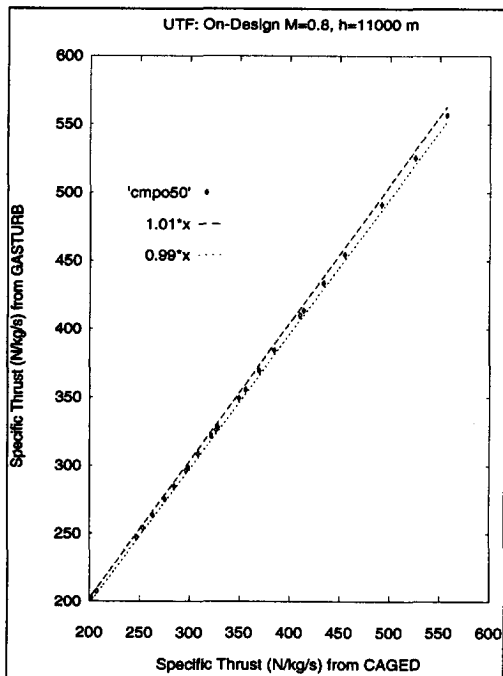
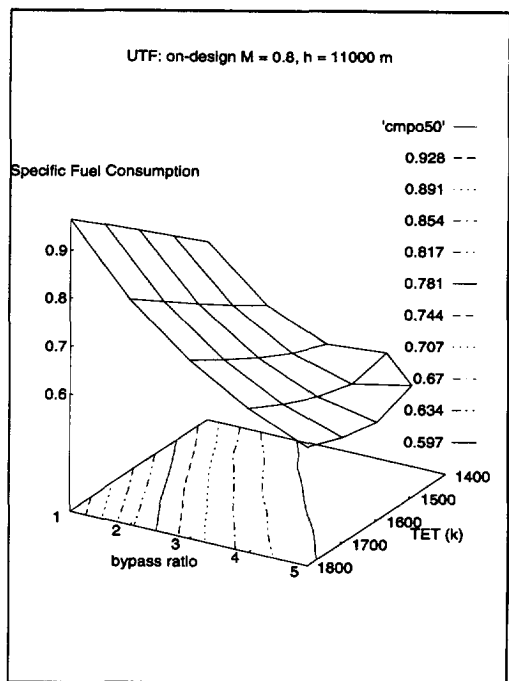
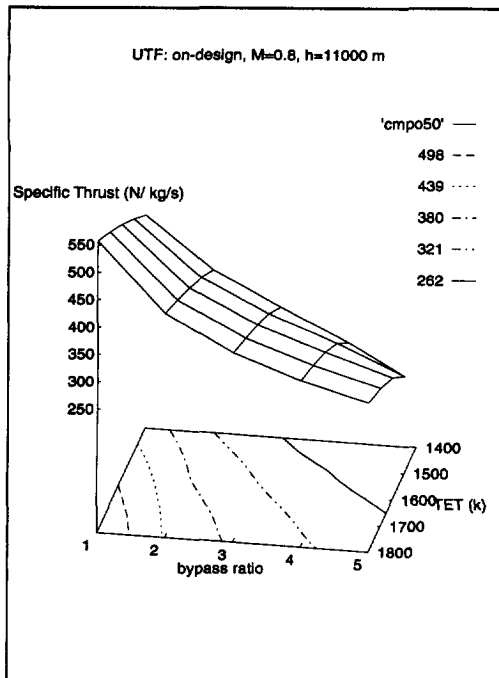


FIG. 53 On-design performance of unmixed turbofan at $M = 0.8$, $h = 11000$ conditions. Major cycle parameters: overall pressure ratio = 50, fan pressure ratio = 3.0, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

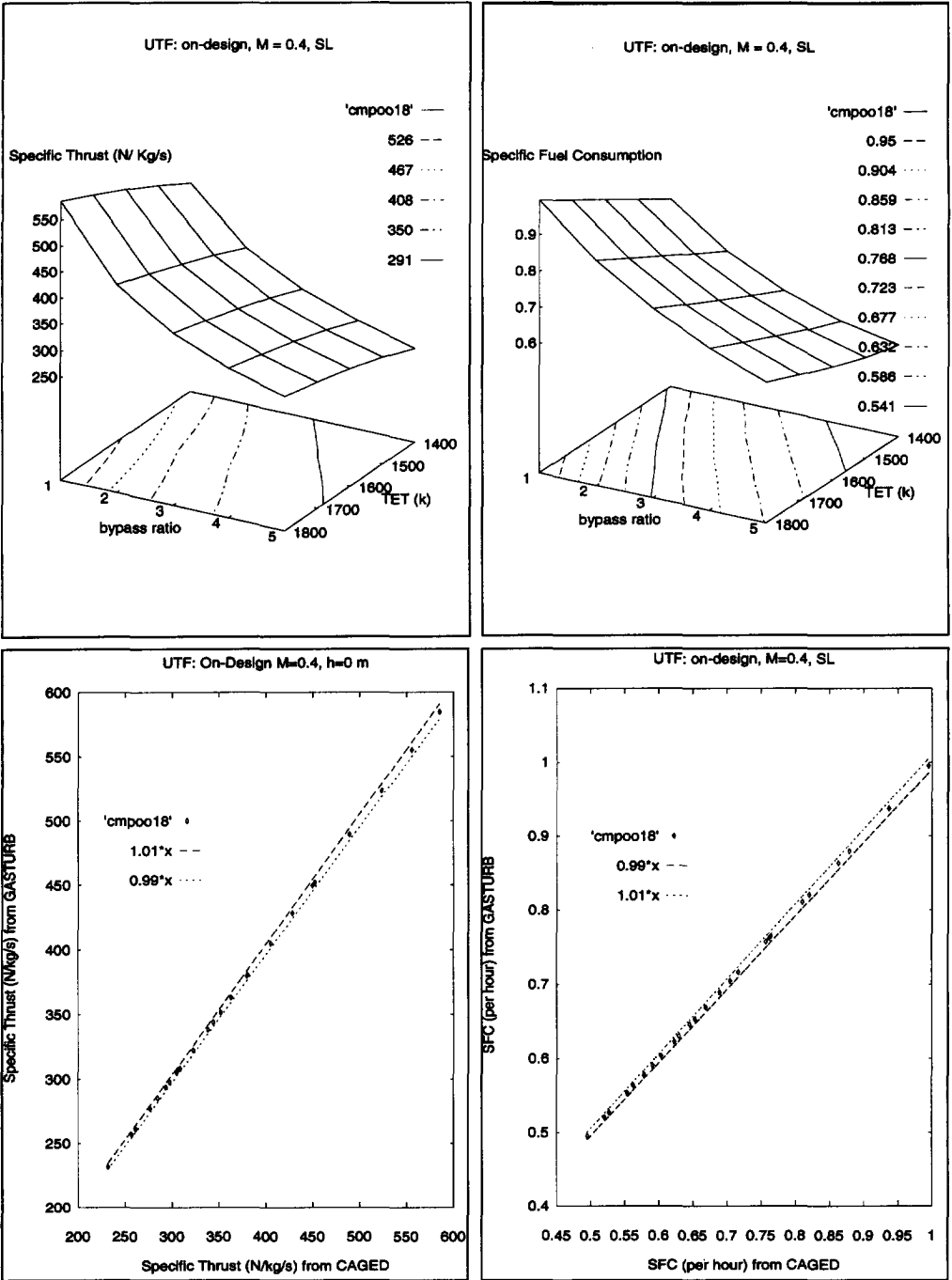


FIG. 54 On-design performance of unmixed turbofan at $M = 0.4$, SL conditions. Major cycle parameters: overall pressure ratio = 18, fan pressure ratio = 1.8, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

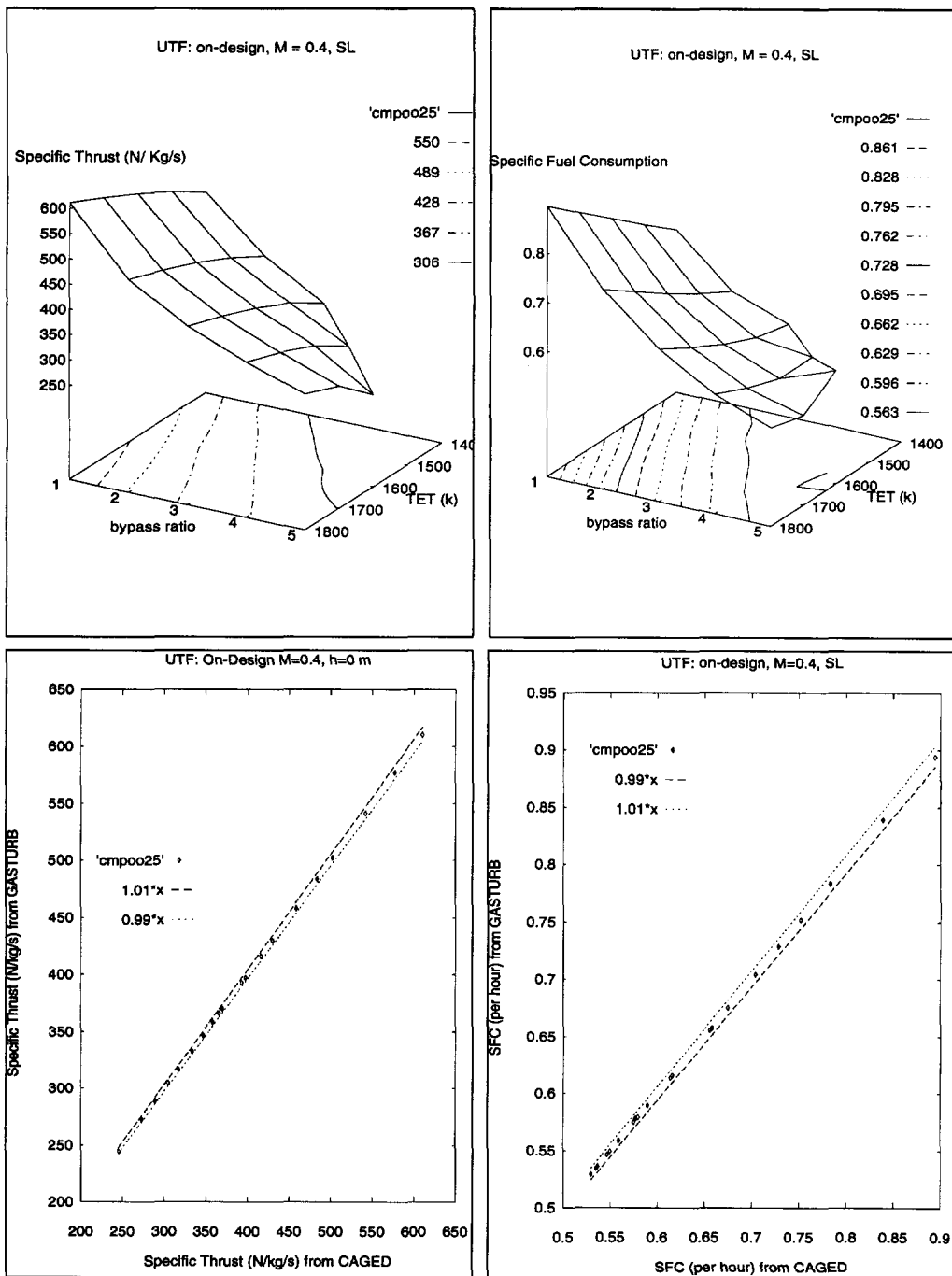


FIG. 55 On-design performance of unmixed turbofan at $M = 0.4$, SL conditions. Major cycle parameters: overall pressure ratio = 25, fan pressure ratio = 2.5, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

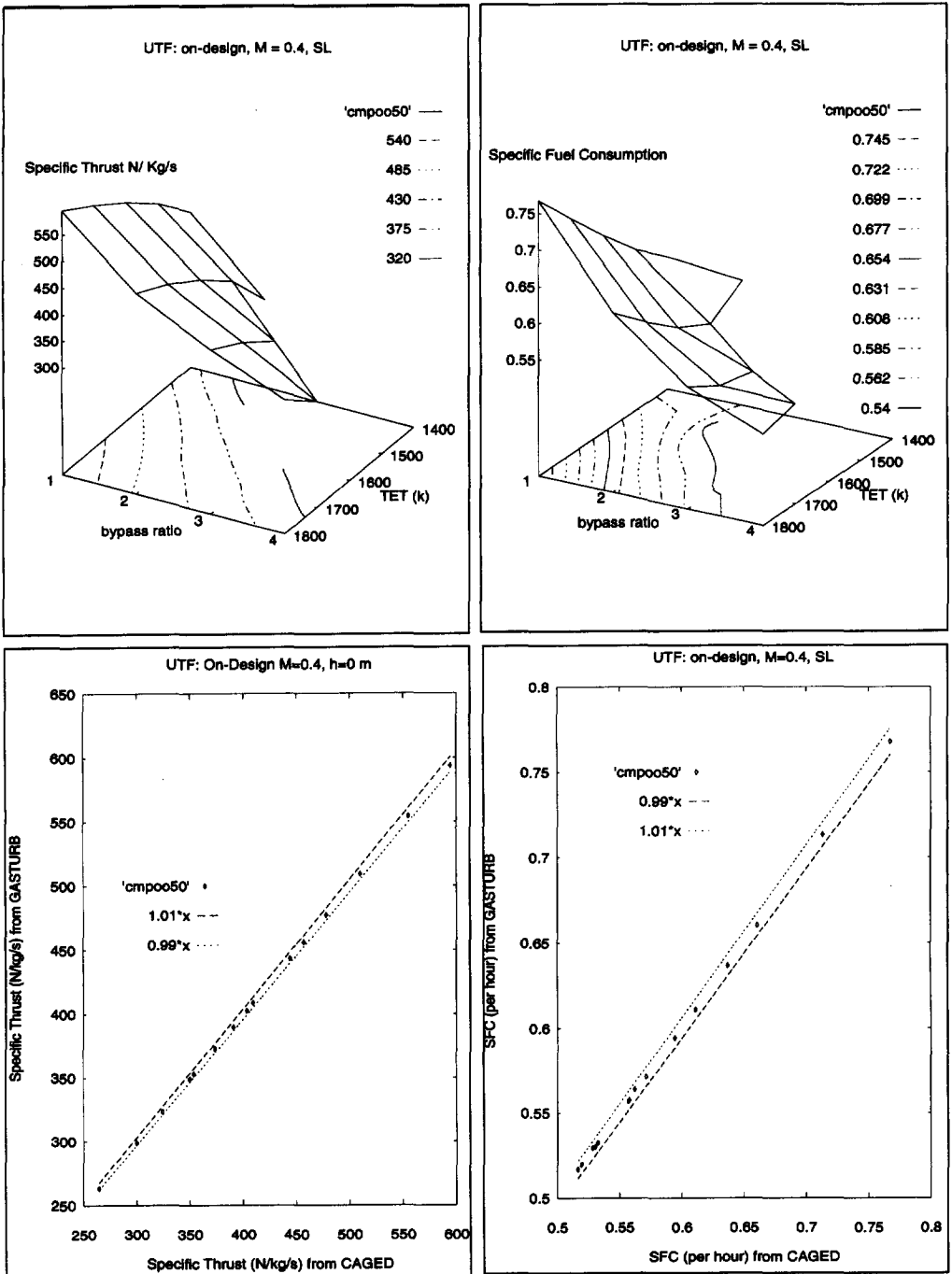


FIG. 56 On-design performance of unmixed turbofan at $M = 0.4$, SL conditions. Major cycle parameters: overall pressure ratio = 50.0, fan pressure ratio = 3.0, TET = 1400 to 1800 °K, bypass ratio = 1 to 5.

In the surface plots, those points which represent infeasible cycles have been omitted making the surface look chopped or clipped. For example in the top two plots of *Fig. 53*, the combination of $TET=1400$ and $Bypass=5.0$ represents an infeasible point and has therefore been clipped or chopped. The feasibility criteria used in these plots is that the static pressure at the exit of exhaust nozzle should not be less than the ambient pressure.

All the comparison plots show that both SFC and F_s data points are placed in the middle of the $\pm 1\%$ band. In fact the data from the two programs are on average not more than 0.1 to 0.2% different. This suggests that the calculated properties of the working gas (i.e. pressure & temperature) at the various engine stations are practically the same between the programs. For example the maximum disagreement in gas temperature was less than $0.5^\circ K$ which occurred at hp-compressor exit station. The calculation of thermodynamic processes are not identical between the two programs which causes the small difference in results.

The practical equivalence of *CAGED* and *GASTURB* on-design results suggest that if the latter is considered as rigorous, so is the former. However the former produces data of the same quality at several orders of magnitude higher speeds *and* the same can be done for *any* engine configuration thanks to the generalized approach.

The on-design performance calculations are significantly sensitive to the accuracy of calculation of gas properties (e.g. C_p , h_g etc.). For this reason, very accurate thermodynamic codes (e.g. *NEPEQ*^[17]) attempt to solve the chemical equilibrium calculations at every station. Apart from accuracy, another advantage of this approach is that any working gas (i.e. a combination of multiple fuels and multiple oxidants) can be handled in the engine models. The disadvantage is that the engine models become extremely sluggish and therefore not acceptable in *CAGED*.

So is it possible to use very accurate gas properties in engine models without adversely affecting their high execution speeds? The answer is yes. It is possible to run a chemical equilibrium code (e.g. *CEC71*^[25]) for a given set of fuels, oxidants, mixture ratios, pressures and temperatures, only one time in advance. This results in a multi-dimensional data table. For example, one fuel and one oxidant would result in a three dimensional data table with pressure, temperature and equivalence ratio as the independent parameters of the table. All that remains now is to generate gas properties from this data table in real time (i.e. at very high speed). The *Modified Multi-variate Newton Raphson* scheme (see section 6.6.1) was developed for this purpose. All the necessary modules for high speed generation of multi-dimensional function from data tables were programmed and it was found that up to 5 dimensional tables (i.e. allows three reactants) could be generated at sufficiently high speed to be suitable for

generating gas properties in *CAGED* engine models. Unfortunately there was no time for the little bit of work necessary to implement this facility in *CAGED* engine models.

Until such time that generation of gas properties from data tables are implemented, *CAGED* uses simple polynomials for standard fuel and air. Initially the polynomials from reference^[67] were used and the first attempt at comparison of on-design performance between *CAGED* and *GASTURB* resulted in an average difference of 2% and 4% in F_s and SFC respectively. Further investigation revealed that *GASTURB* uses different set of polynomials^[44] for generating gas properties (the polynomials which caused the unacceptable difference).

Since *GASTURB* was being used for validation of *CAGED* results, it became necessary to implement the same gas properties polynomials^[36] in *CAGED* which resulted in the fantastic agreement as shown in *Fig. 48* up to and including *Fig. 56*. There is a high degree of confidence in the gas properties polynomials used by *GASTURB* because its output compares very well with the *MTU MOPS* program which gets its gas properties from a rigorous source^[25].

The conclusion here is that the on-design thermodynamic state of the engine models are quite sensitive to the gas properties model used and it is essential to use the highest accuracy possible. The high speed data generation technique developed here (The *Modified Newton Raphson*) makes it possible to use very accurate gas properties data from *CEC* programs at no significant penalty to the execution speed of the engine models.

7.2 Validation of Off-design Engine Models

As explained previously, the structure of off-design engine models generated by *CAGED* (Figure 25 on page 83) looks very similar to that of on-design engine models (Figure 19 on page 49) with the exception that some key spool parameters are determined from the closed form solutions derived previously (See section 4.4). In general if the following three spool parameters are known at an off-design point,

- the mass ratio term (or the bypass term) from the ∇m_{rat} term
- the compressor (or fan) pressure ratio from the $\nabla \pi_c$ term
- the temperature ratio \varnothing from the $\nabla \varnothing$ term

then sufficient information is available to calculate the performance of the spool in exactly the same way as in the on-design case, since the sequence and type of thermodynamic processes are identical between the on- and off-design cases.

Now since the high accuracy (~99.9%) and high speed (~millisecond) of on-design engine models has already been illustrated, it remains to be shown that the spool parameters can be determined accurately from the closed form equations. Their closed form and simplicity make it rather obvious that their use will not significantly hamper the speed of execution.

The test engine for comparison between *GASTURB* and *CAGED* is defined in *TABLE 3*

First the high pressure spool will be discussed and it will be shown that we can obtain accuracies of 96% at part-load to 99% at full load by using the *restricted* solution. Again restriction does not automatically imply a choked turbine. All that is implied is that the variation of the corrected mass flow parameters at inlet (μ_3) and exit of the turbine (μ_4), with respect to μ_1 , are small compared to other terms in *eq. 12* and *eq. 19* respectively. This condition typically occurs in high and intermediate pressure spools (see *Fig. 57* and *Fig. 58*).

For the low pressure spool however, the $\nabla\mu_4$ term is not always negligible in *eq. 19*, as illustrated in *Fig. 60*. But the $\nabla\mu_3$ term is still negligible in *eq. 12* for the majority of the *ERL* (see *Fig. 59*).

Then the *unrestricted* solution is tested on the low pressure spool and two problems are identified,

1. A post-correction for the variation of compressor efficiency (the ∇a_c term in *eq. 98*) does not deliver sufficient accuracy for the lp spool, so it has to be incorporated in the derivation of the original equations
2. The assumption that $\nabla(p_{out}/p_{amb})$ is negligible in *eq. 46* gives unreasonable results for the case where the core nozzle is "super choked", i.e. a nozzle that operates at much higher pressure ratios than the critical choking pressure ratio. This assumption is consistent with that of *Wittenberg*^[68] where it did not seem to adversely affect the results. However, the bypass engines (e.g. the RB211 engine) calculated in that reference did not have a super choked nozzle.

TABLE 3

Design point parameters of a two spool unmixed turbofan test engine

<i>Parameter</i>	<i>Values</i>
intake polytropic efficiency	1.00
fan pressure ratio	1.8
overall pressure ratio	18
fan polytropic efficiency	0.9079
compressor polytropic efficiency	0.9044
burner pressure ratio	0.97
burner combustion efficiency	0.99
bypass ratio	2.0
turbine entry temperature	1800 °K
hp-turbine polytropic efficiency	0.91
lp-turbine polytropic efficiency	0.92
core nozzle polytropic efficiency	1.00
bypass nozzle polytropic efficiency	1.00
inter-duct pressure ratios	1.00
bleeds	0.0 both spools
mechanical efficiency	1.00 both spools
shaft power off-take	0.0 both spools
altitude	0.0 m
flight mach number	0.0
nozzle applied pressure ratio $P_{t_{out}}/P_{amb}$	4.8974 (core), 1.80 (bypass)

The first problem above was relatively easy to fix, by re-deriving the equations and keeping the ∇a_c term in the analysis, as explained in the next section. The second problem above required a special derivations specially for choked and super choked core nozzle. See section 7.2.3 for further discussion on this issue.

7.2.1 Modification to the original closed form equations

The accuracy of the original *restricted* and *unrestricted* forms can be improved by using the identity,

$$\tau_c = \pi_c^{1/a_c} \Rightarrow \nabla \tau_c \equiv \frac{\nabla \pi_c}{a_c} - \nabla a_c \times \ln(\tau_c) \quad (191)$$

instead of the approximation that was originally used,

$$\nabla \tau_c = \frac{\nabla \pi_c}{a_c} \quad (192)$$

Using *eq. 191* and following exactly the same procedure as in section 4.4.1.4, then the two equations for $\nabla \tau_s$ become,

$$\begin{aligned} \frac{\nabla \tau_s}{2} = \nabla \pi_s - \nabla m_{rat} - 1 = \nabla \pi_c k \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} \left(1 - \frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \right) + 1 \right) \\ + \nabla \pi_b k \left(\frac{2a_t(\varnothing - \tau_s)}{\tau_s} + 1 \right) + \nabla m_{rat} \left(- \frac{a_t k (\varnothing - \tau_s)}{\tau_s} - 1 \right) + \\ - 1 - \frac{2a_t k (\varnothing - \tau_s)}{\tau_s} \left(1 - \nabla a_c \times \frac{\ln(\tau_c)}{2} \frac{\tau_c}{\tau_c - 1} \right) \end{aligned} \quad (193)$$

and,

$$\begin{aligned} \frac{\nabla \tau_s}{2} = \frac{\nabla \pi_c}{\tau_s} \left(\varnothing - \frac{\tau_c}{2C_{p_{rat}} m_{rat} a_c} \right) + \frac{\varnothing}{\tau_s} (\nabla \pi_b - 1) + \\ + \frac{\nabla m_{rat}}{\tau_s} \left(-\varnothing + \frac{\tau_c - 1}{2C_{p_{rat}} m_{rat}} \right) + \frac{1}{\tau_s} \left(\nabla a_c \times \ln(\tau_c) \frac{\tau_c}{2C_{p_{rat}} m_{rat}} \right) \end{aligned} \quad (194)$$

where the above two equations are the new versions of *eq. 20* and *eq. 23* respectively. Equating the two equations and manipulating them exactly as in section 4.4.1.4 gives the more accurate version of the *unrestricted* solution,

$$\nabla \pi_c = \frac{1}{B+C} \left(1 + \nabla \pi_b (-1-C) + \frac{\nabla m_{rat}}{2} + \right. \\ \left. + a_r (1-k) (\nabla \tau_r) + -\nabla a_c \times \frac{\ln(\tau_c)}{2} \frac{\tau_c}{\tau_c - 1} \right) \quad (195)$$

where,

$$\left\{ \begin{array}{l} B = 1 - \frac{\tau_c}{2a_c (\tau_c - 1)} \\ C = \frac{k-1}{(2a_t k - 1) \left(\frac{\emptyset - \tau_s}{\tau_s} \right)} \\ k = 1 - \frac{1}{\nabla_{\mu_{in}} \pi_n} \end{array} \right. \quad (196)$$

c.f. eq. 52. The *restricted* form is obtained simply by substituting infinity for $\nabla_{\mu_{in}} \pi_n$, which results in $k = 1$ and $C = 0$.

Note that the above solution does not require a two step correction any more. But it does still require one correction for the shaft power off-take as before.

7.2.2 The High Pressure Spool Calculated with the *Restricted Outlet* Solution

The accuracy of the results for the high pressure spool were very encouraging indeed for a purely closed form analytical approach. This is partly due to the fact that the high pressure spool is in fact very much in line with what the author has termed the *restricted* spool, and partly due to the fact that *GASTURB* assumes a constant corrected mass flow at the high pressure turbine inlet, *ensuring* that all assumptions are exactly correct.

Fig. 61 up to and including *Fig. 71* compare the *ERL* hp- compressor pressure ratios as calculated by the *restricted* form solution and *GASTURB*. The result of the latter is generally plotted on the X- axis and the former on the Y- axis. Most of the plots include a $Y = 0.95 X$, $Y=X$ and $Y=1.05 X$, which represent the -5%, 0% and 95% accuracy respectively.

Fig. 61 plots compares the compressor pressure ratios π_c and shows that the error is very small near the design point and accumulates to about 4% at the lowest part of the curve. This accumulation of error was anticipated and discussed earlier. Basically the solution starts at the design point (or some other known reference point on the *ERL*) and there will be a very small error in the position of the next point on the *ERL* after extrapolation. Now since the updated gradients at this new point do not exactly fall on the *ERL* and since the new value of $\nabla\pi_c$ is sensitive to the exact position of this point, then the next point will be even farther from the *ERL* and this process leads to an accumulation of errors. There can be no physical reason for the accumulation of errors in this particular case, because as mentioned above, *GASTURB* assumes constant corrected mass flow at entry to the hp- turbine. So it is not correct to say that the accumulation of errors may be due to the assumptions becoming less true as we move down the *ERL*.

The calculation of the ϕ parameter also compares very well as shown in *Fig. 62* with similar level of accuracies. It should be mentioned at this stage that the variation of the mass ratio has been set equal to zero. This is not strictly true since the amount of fuel flow is a function of ϕ and π_c which varies along the *ERL*. *Cockshutt*^[7] derives a very elegant formula for the gradient of the fuel air ratio which could have easily been incorporated in the closed form solutions above. However test runs indicated that the key spool gradients, $\nabla\pi_c$ and $\nabla\phi$, are not significantly sensitive to the variation of fuel air ratio and there was not much to be gained by the added complexity of incorporating them into the solutionⁱ. It should be clear to the reader that we are not setting the fuel to air ratio to be constant, on the contrary. A correct value of f is calculated from the burner enthalpy equation, after every new point on the *ERL* is found.

The compressor temperature ratio τ_c can be calculated much more accurately than pressure ratio (*Fig. 63* shows a maximum error of around 1% at part load) because the $\nabla\tau_c$ term is much smaller than the $\nabla\pi_c$ term. In other words a small error in $\nabla\tau_c$ is magnified since $\nabla\pi_c = a_c \nabla\tau_c$ and a_c is about 3.5 for 100% compressor polytropic efficiency.

i. the variation of fuel to air ratio may become significant in some special engine concepts where the turbine entry temperature goes through a rapid change along the *ERL*.

Fig. 64 shows a very curious curve of \varnothing vs. π_c plotted on the X- and Y- axis respectively. The solid curve through filled dots are from *GASTURB* and the *restricted* form is represented by pluses. It is very clear to that the position of *GASTURB* points are not identical to the position of *CAGED* points at part load, but for some reason unknown to the author, of all the directions along which a shift could take place, it takes place exactly along the curve. Unfortunately there was no time to find out whether this is a “freak” or a reproducible result for any restricted spool. If it is repeatable then it can be used to an advantage leading to a conclusion that it is better to get the value of \varnothing from π_c or vice versa by interpolation of the curve rather than taking the value that comes out of the extrapolations directly. The curve of \varnothing vs. π_s in Fig. 69 exhibits a similar shift too.

The variation of \varnothing and π_c as a function of the mass flow parameter at the inlet to the compressor (μ_1) are plotted in Fig. 65 and Fig. 66 respectively. The solid lines and the dashed lines indicate the rigorous and the *restricted* solution respectively. Again the agreement is very good.

Fig. 70 shows at first sight an unbelievable agreement between the calculated and rigorous burner pressure ratios. The maximum error is around 0.2% at part-load. More careful reading of the burner section of the *GASTURB* manual revealed that the two actually use the same model for burner pressure ratio, only that we have obtained a differential form of the same pressure loss model. So there is no major conclusion to be drawn from this figure. The results from the closed form *restricted* solution is shown to be in good agreement with the more rigorous results from *GASTURB*.

Fig. 71 summarizes the results by plotting all high pressure spool parameters of interest versus the mass flow parameter at the compressor inlet.

Now Wittenberg^{[68] [69]} also gives a closed form solution for the high pressureⁱ spool, and demonstrated the high quality of his solution by comparison with real engine data for several engine types. The following questions may arise in the reader's mind,

- i. In his paper, he also mentioned that his closed form solution can be extended to treat a low pressure spool with bypass, for the special case of choked hot and cold nozzle, but he omitted it because he thought that the resulting complicated expression will not be of general interest

- if his solution is so good, why bother with the heavy analytical work in the present thesis?
- to what extent is *Wittenberg's* result similar to the closed differential forms derived in the present thesis?

It was not necessary to include *Wittenberg's* results in the plots discussed in this section because an analytical comparison is possible which is quite straightforward as follows.

Let us begin with the two *Wittenberg* equations that together define the closed form solutions for the high pressure spool with choked turbine,

$$\mu_1 = \mu_{1_{design}} \frac{1}{\sqrt{\emptyset}} \frac{\pi_c}{\pi_{c_{design}}} = k_1 \frac{\pi_c}{\sqrt{\emptyset}} \quad (197)$$

$$\pi_c = \left[1 + \emptyset \left(\pi_{c_{design}}^{1/a_c} - 1 \right) \right]^{a_c} = [1 + \emptyset k_2]^{a_c} \quad (198)$$

Where k_1 and k_2 are function of design point parameters and are constant. The above equations are exact copy of equations 4.7 and 4.5 of reference 68 except that the symbols have been made consistent with our nomenclature.

One can immediately observe that for any given value of \emptyset , μ_1 and π_c can be directly determined from the above two equations, hence the claim that they are closed. Now is this result usable in a general network where the flow of information is towards downstream? The answer is no, because in a general network, with forward flowing information, the components that are placed along a single aerodynamic stream, can only communicate by passing values of μ_1 to the next downstream component. In other words for any given new value of μ_1 , we do not automatically know the value of \emptyset to make the closed form solution possible. Now let us see if we can manipulate *Wittenberg's* result into a closed form that starts by using a value of μ_1 instead of \emptyset .

Substituting for \emptyset from eq. 197 into eq. 198, gives,

$$\pi_c = \left[1 + k_1^2 k_2 \frac{\pi_c^2}{\mu_1^2} \right]^{a_c} \quad (199)$$

and rearranging, finally yields,

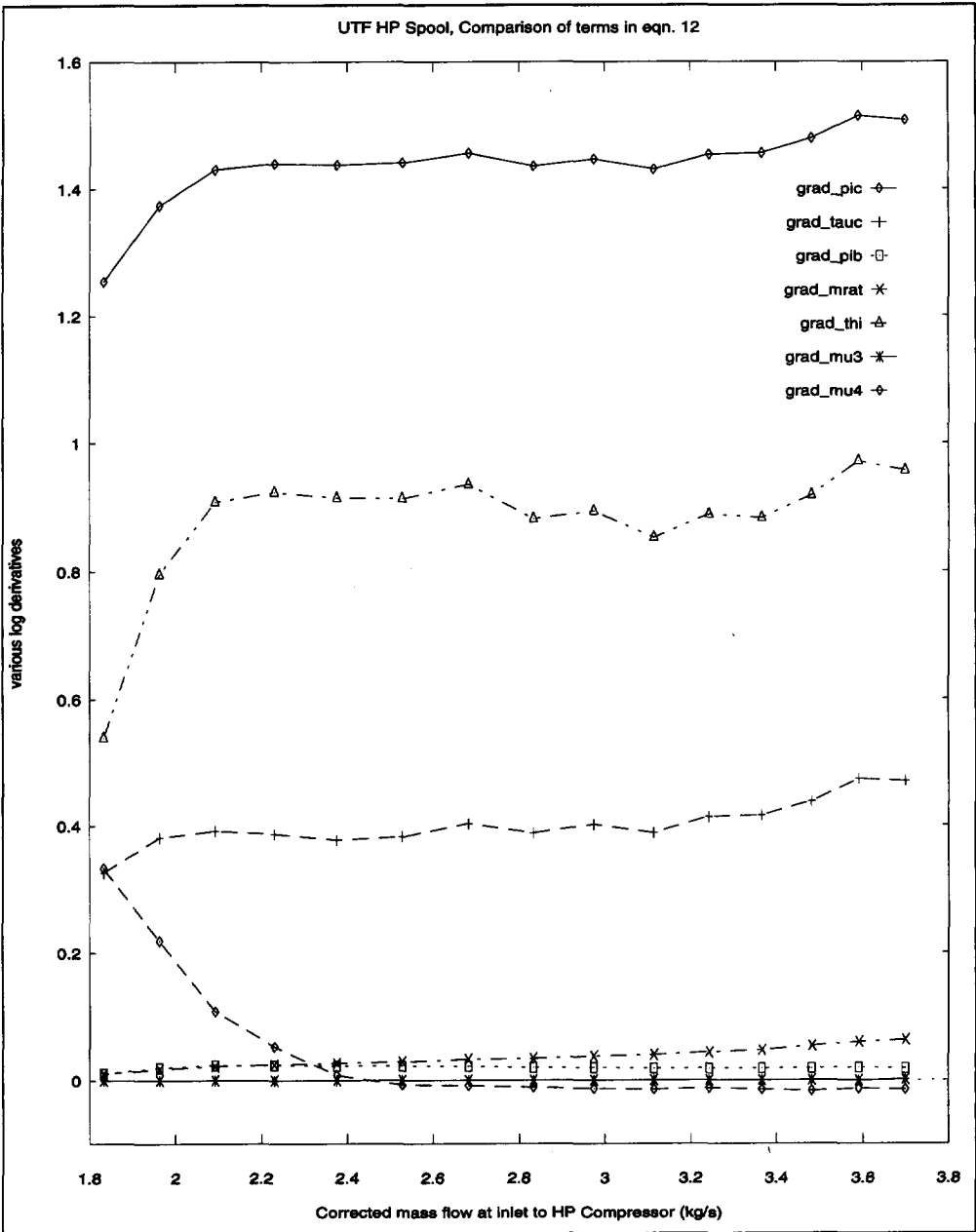


FIG. 57 Magnitude comparison between the terms that appear in eq. 12 for the hp-spool, as calculated by GASTURB. This figure shows that the variation of corrected mass flow at inlet to hp-turbine is negligibly small compared to the other terms in that equation.

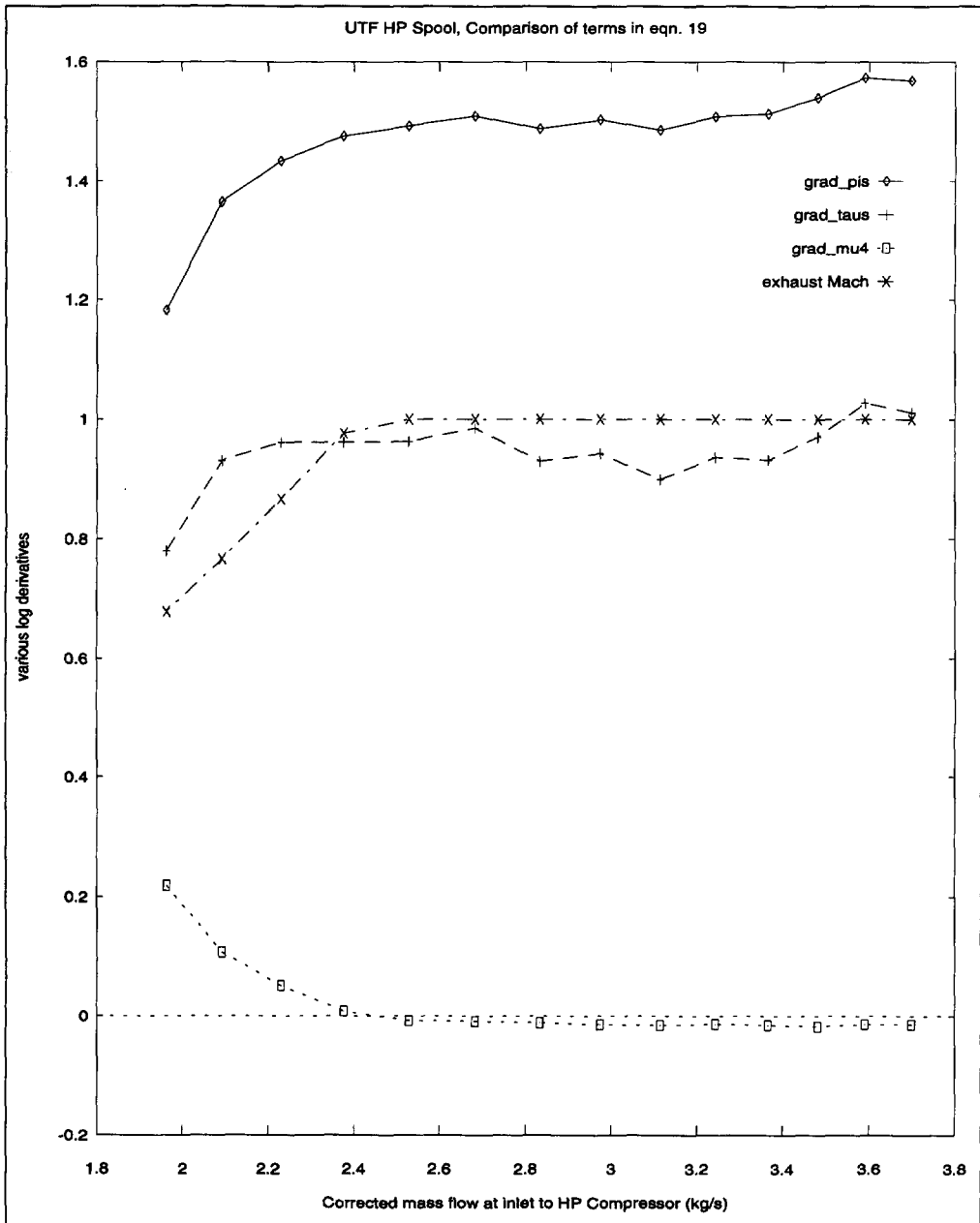


FIG. 58 Magnitude comparison between the terms that appear in eq. 19 for the hp-spool, as calculated by GASTURB. This figure shows that for the majority of operating points, the variation of corrected mass flow at outlet of hp-turbine is negligibly small compared to the other terms in that equation.

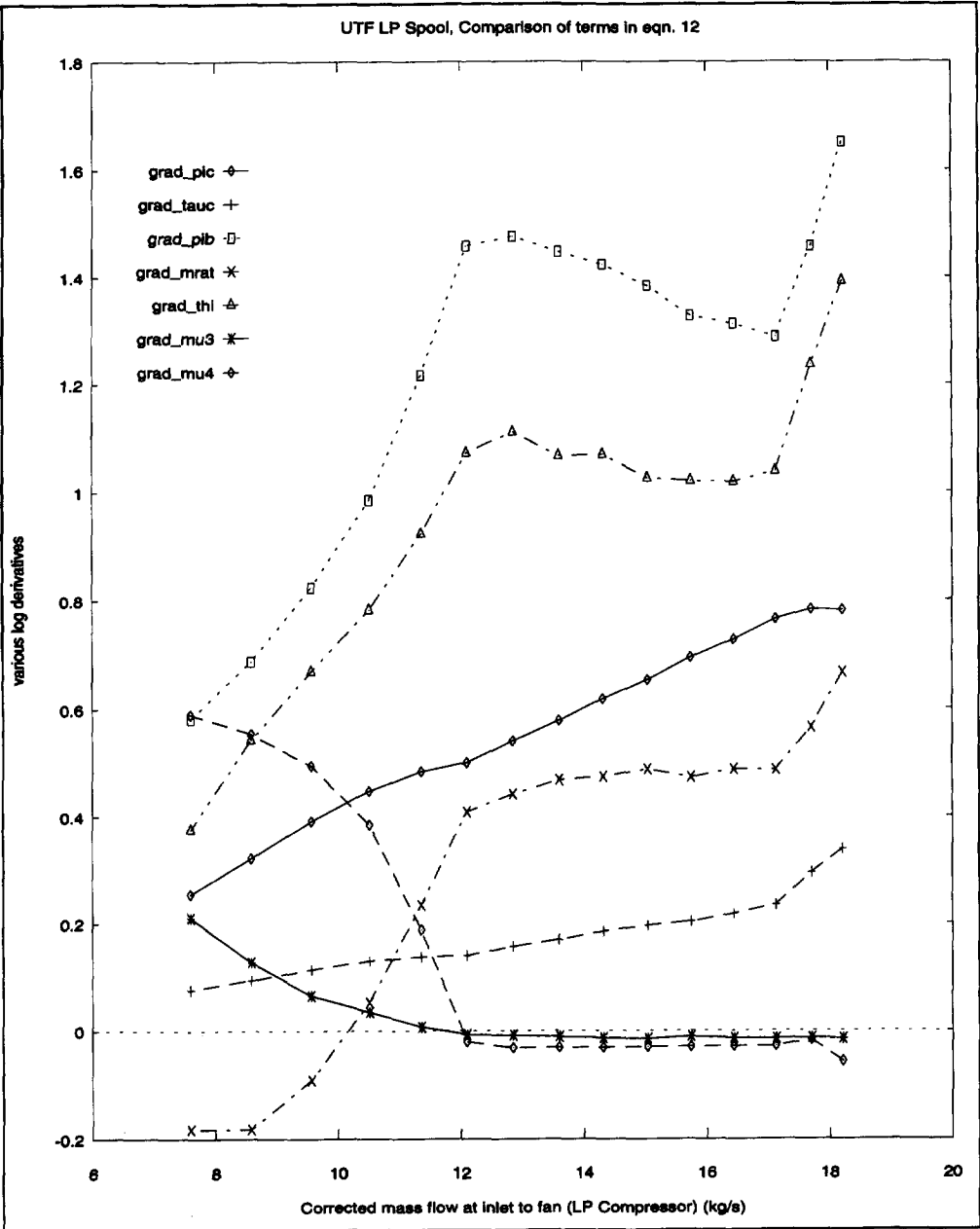


FIG. S9 Magnitude comparison between the terms that appear in eq. 12 for the lp-spool, as calculated by *GASTURB*. This figure shows that for majority of operating points, the variation of corrected mass flow at inlet to lp-turbine is negligibly small compared to the other terms in that equation.

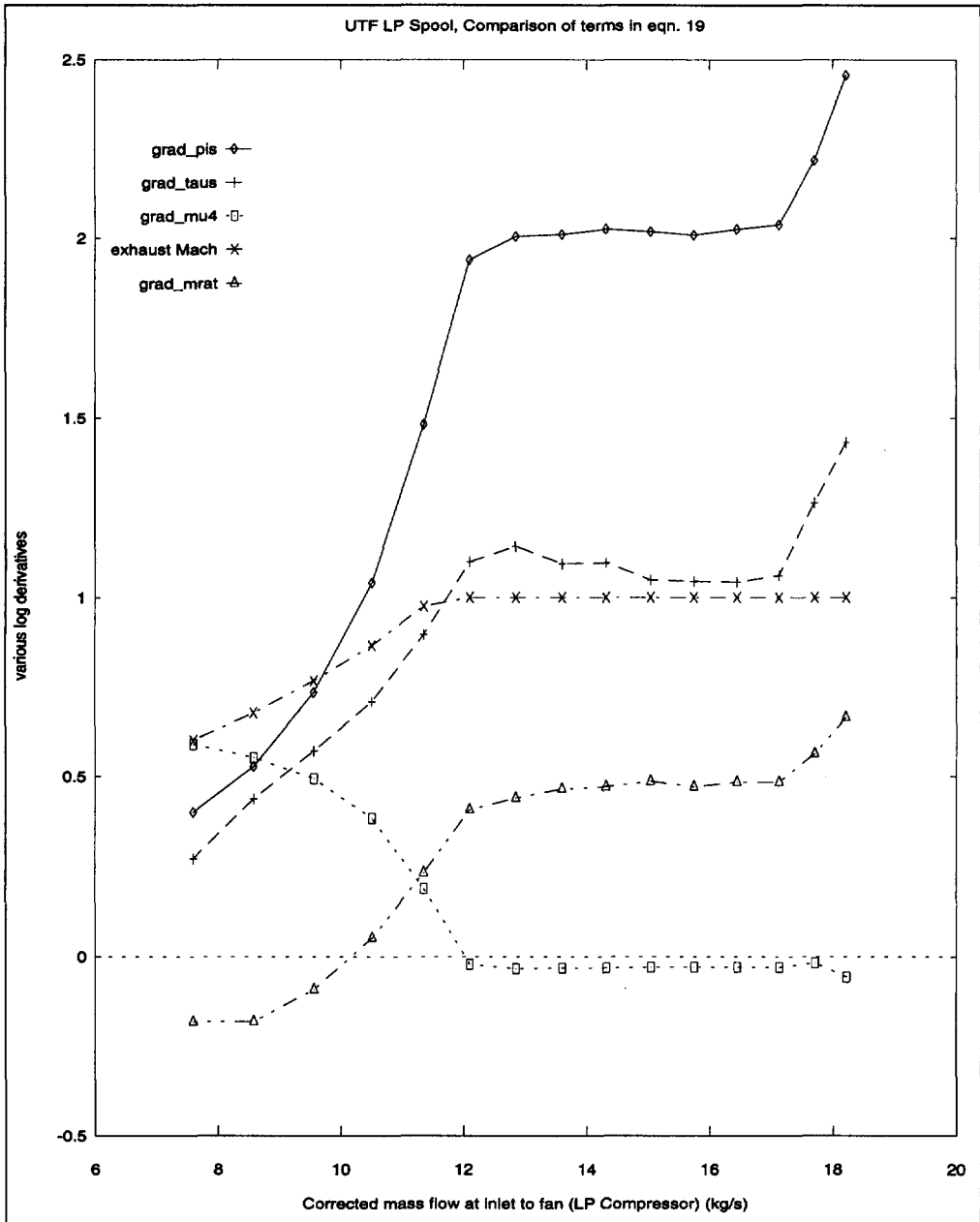


FIG. 60 Magnitude comparison between the terms that appear in *eq. 19* for the lp-spool, as calculated by *GASTURB*. This figure shows that the lp-spool is not in general restricted, and that the variation of corrected mass flow at outlet from the lp-turbine can be significant.

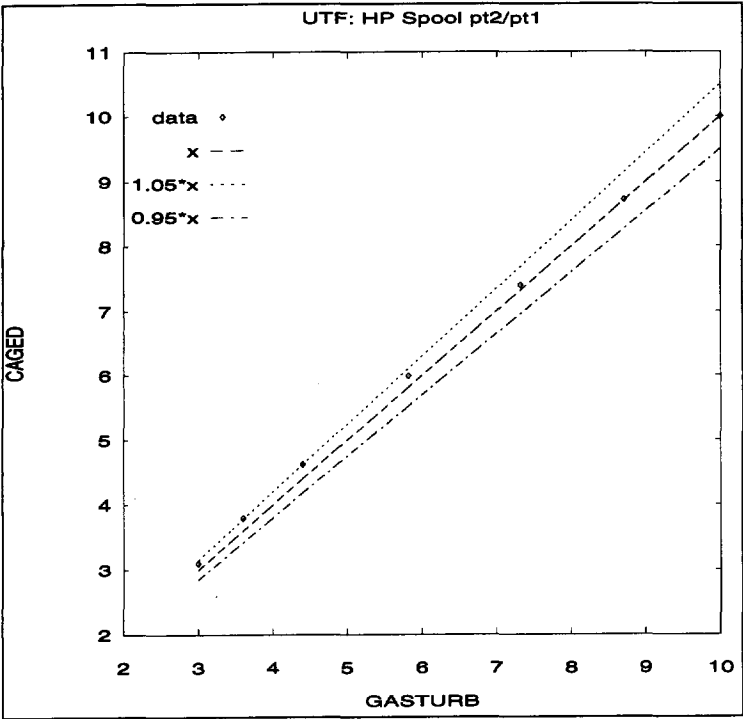


FIG. 61 Comparison of off-design hp-compressor pressure ratios (π_c) for the test engine. The vertical axis represents values from CAGED and the horizontal axis represents values from GASTURB.

FIG. 62 Comparison of off-design hp-spool temperature ratios (ϕ) for the test engine. The vertical axis represents values from CAGED and the horizontal axis represents values from GASTURB.

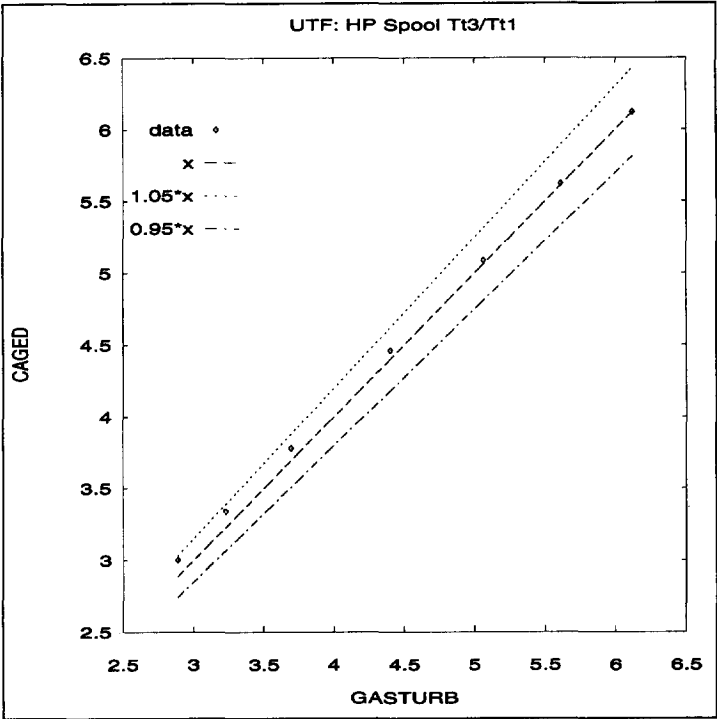


FIG. 63 Comparison of off-design hp-compressor temperature ratios (τ_c) for the test engine. The vertical axis represents values from CAGED and the horizontal axis represents values from GASTURB.

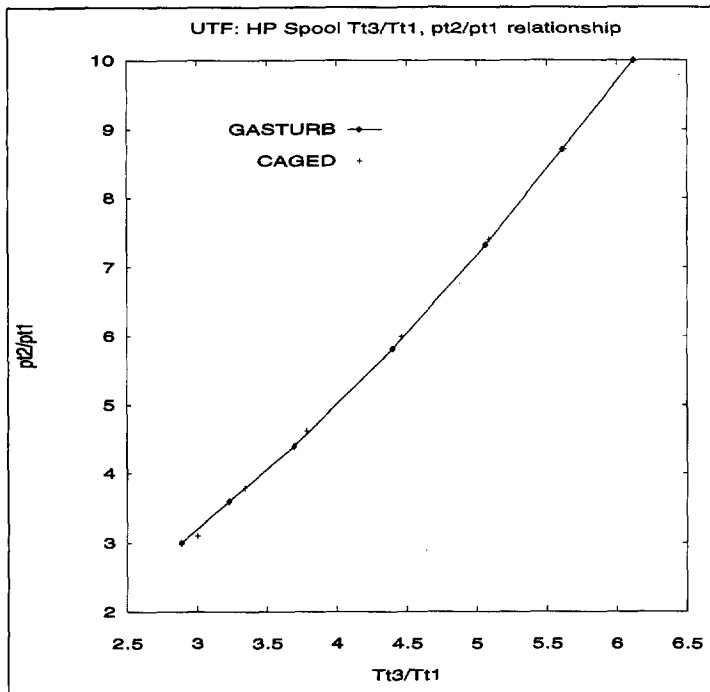
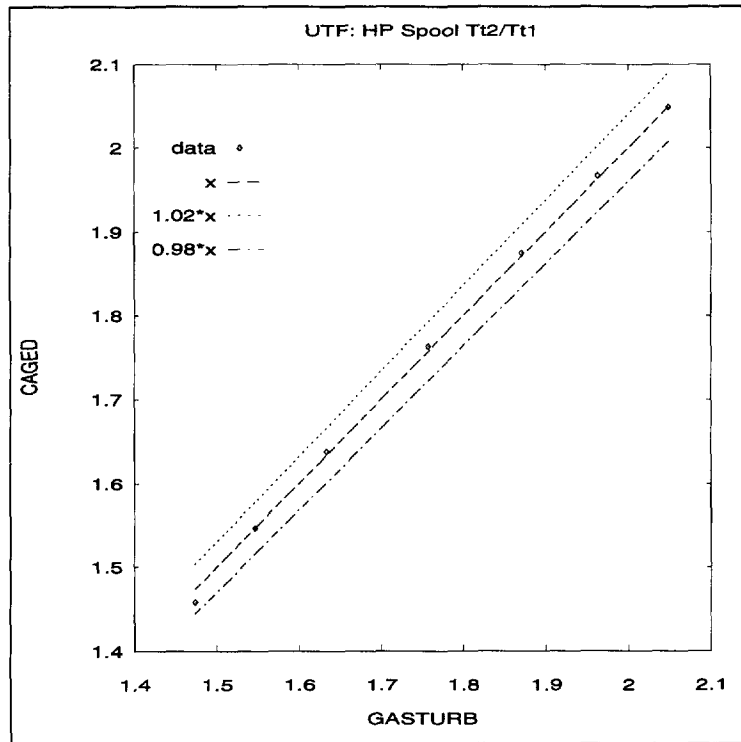


FIG. 64 π_c versus ϕ relationship for the hp-spool of unmixed turbofan, as calculated by GASTURB and CAGED.

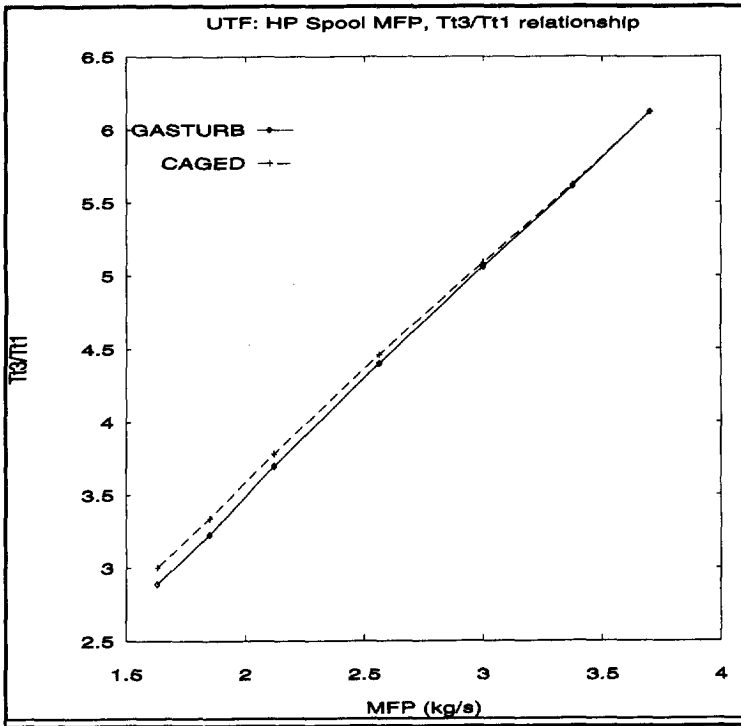


FIG. 65 ϕ versus MFP relationship for the hp-spool of the test engine, as calculated by GASTURB and CAGED.

FIG. 66 π_c versus MFP relationship for the hp-spool of the test engine, as calculated by GASTURB and CAGED.

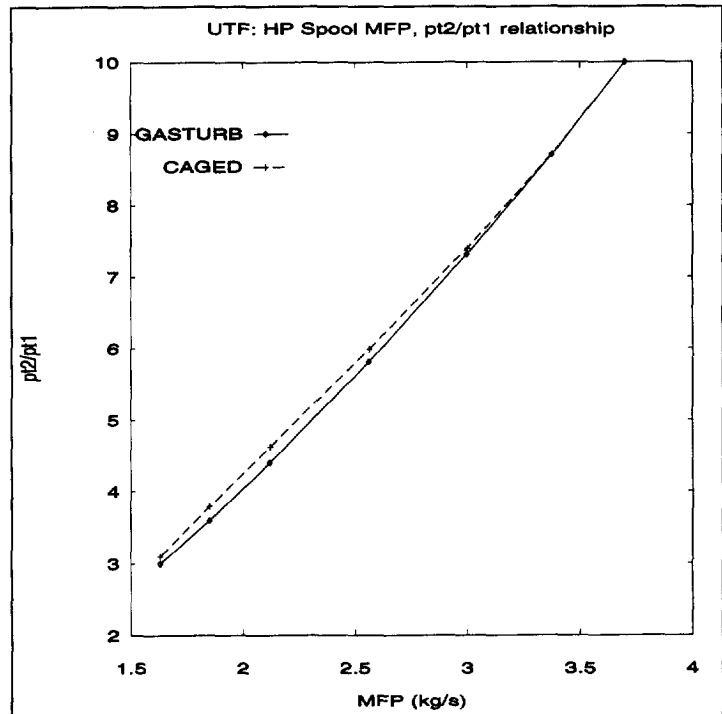


FIG. 67 Comparison of off-design hp-spool pressure ratios (π_s) for the test engine. The vertical axis represents values from CAGED and horizontal axis represents values from GASTURB.

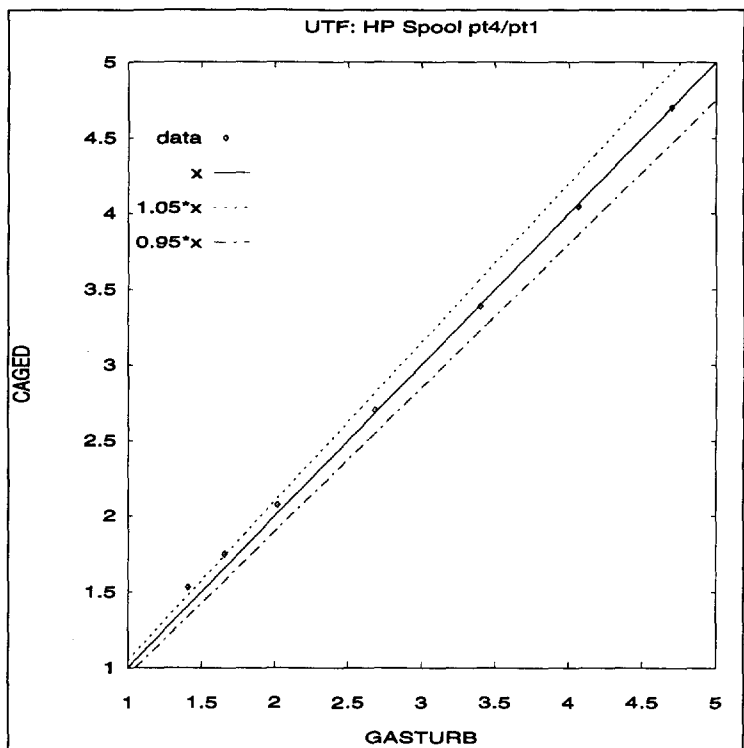
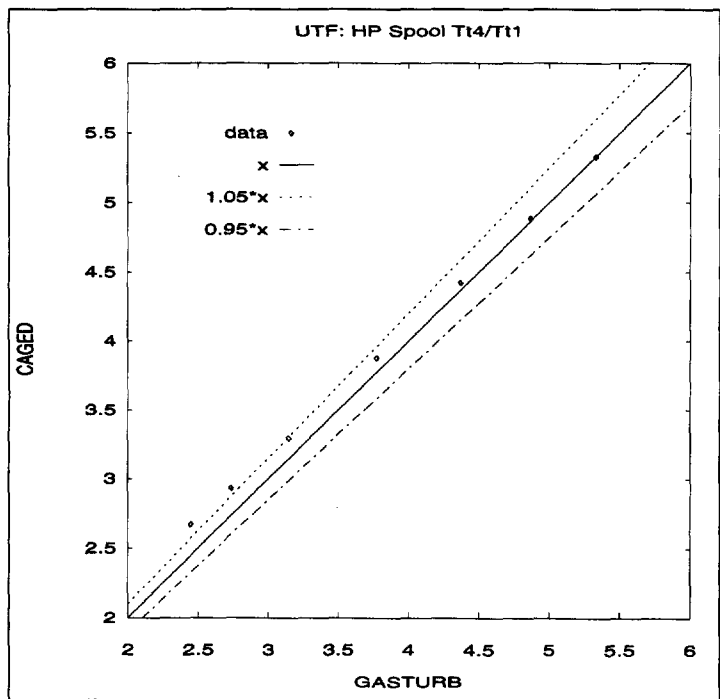


FIG. 68 Comparison of off-design hp-spool temperature ratios (τ_s) for the test engine. The vertical axis represents values from CAGED and the horizontal axis represents values from GASTURB.



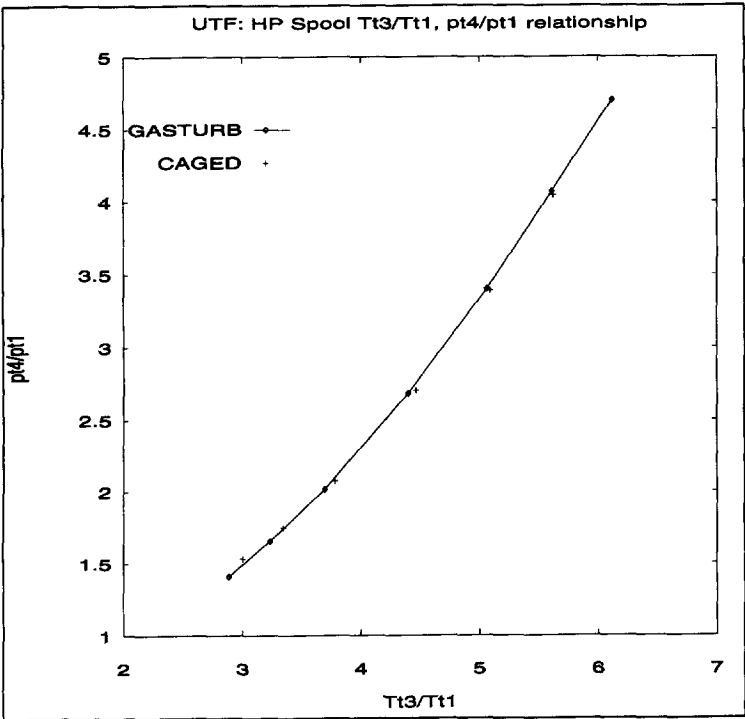
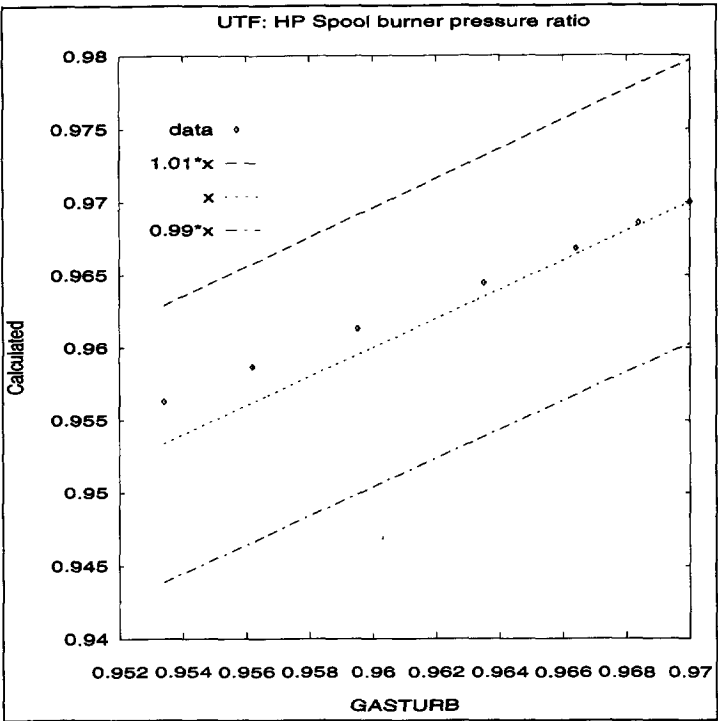


FIG. 69 π_s versus ϕ relationship for the hp-spool of the test engine, as calculated by *GASTURB* and *CAGED*.

FIG. 70 Comparison of off-design burner pressure ratios (π_b) for the test engine. The vertical axis represents values from *CAGED* and horizontal axis represents values from *GASTURB*.



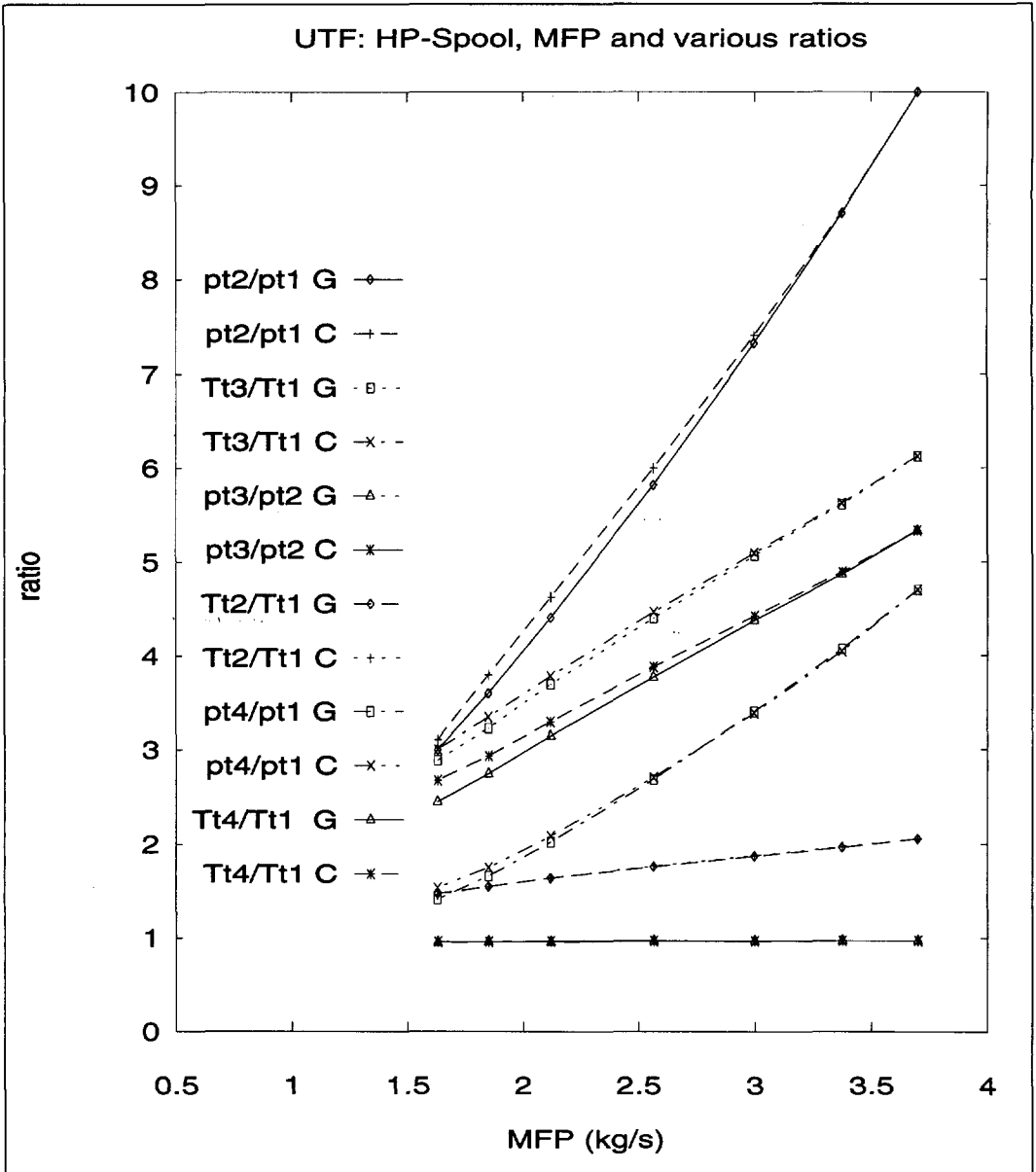


FIG. 71 Collection of curves of off-design hp-spool parameters for the test engine as calculated by *CAGED* (designated C) and *GASTURB* (designated G).

$$\pi_c^{1/a_c} - (k_1^2 k_2) \frac{\pi_c^2}{\mu_1^2} - 1 = 0 \quad (200)$$

The above equation would be closed form and applicable to *Sequential Network Logic* if and only if an analytical solution is found for the above equation for any given value of μ_1 . Now the value of a_c is around 3.5 and 3.22 for 100% and 92% polytropic compressor efficiency respectively. Since the value of $1/a_c$ is not a “nice” value (e.g. 0.2857 for the ideal compressor), it is doubtful that an analytical solution can be found for the above equationⁱ.

Let us now apply the ∇ operator to eq. 199 and find out how well *Wittenberg's* method correlates with our results,

$$\begin{aligned} \nabla \pi_c &= a_c \nabla \left[1 + k_1^2 k_2 \frac{\pi_c^2}{\mu_1^2} \right] = \frac{a_c}{\left[1 + k_1^2 k_2 \frac{\pi_c^2}{\mu_1^2} \right]} \left[k_1^2 k_2 \frac{\pi_c^2}{\mu_1^2} \nabla \left(k_1^2 k_2 \frac{\pi_c^2}{\mu_1^2} \right) \right] \\ &= \frac{a_c}{\pi_c^{1/a_c}} \left(\pi_c^{1/a_c} - 1 \right) (2 \nabla \pi_c - 2) \end{aligned} \quad (201)$$

rearranging using $\pi_c = \tau_c^a$ and $\nabla \pi_c = a_c \nabla \tau_c$ finally yields,

$$\nabla \pi_c = \frac{1}{1 - \frac{\tau_c}{2a_c(\tau_c - 1)}} \quad (202)$$

Now remembering *Cockshutt's*^[7] result,

i. However feeding equations to a mathematical package (e.g. *Mathematica*) sometimes leads to surprises and may find an analytical solution to the above equation.

FIG. 72 Modelling the core nozzle applied pressure ratio with the assumption that $\nabla(P_{amb}/P_{out})$ is negligible leads to large errors. Ideally the data points should follow the $Y=X$ line closely. The left part of the curve corresponds to the "super choked" exhaust nozzle.

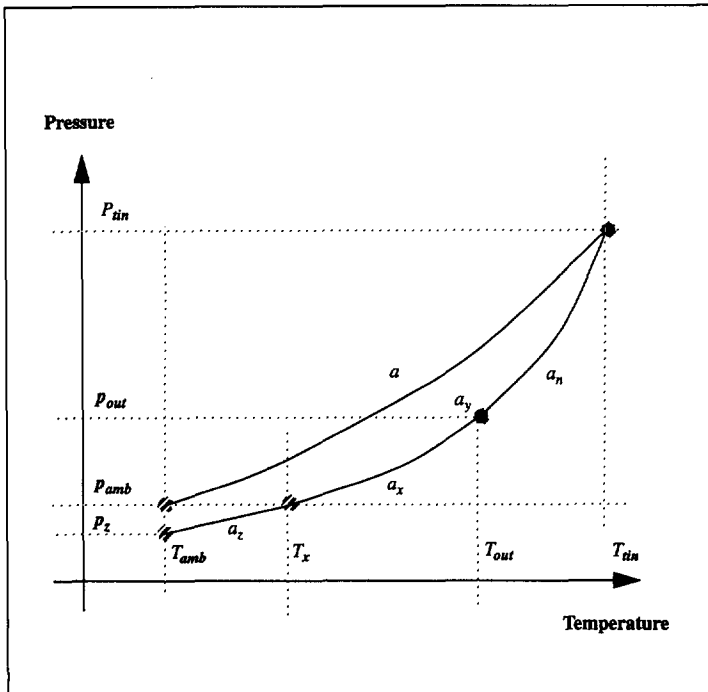
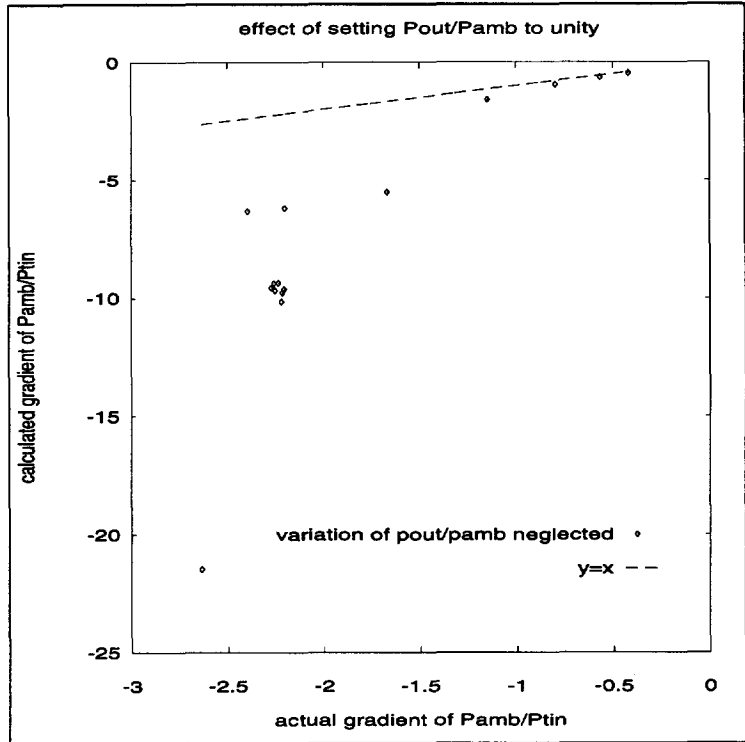


FIG. 73 Break-down of internal and external expansion of core nozzle into thermodynamic processes.

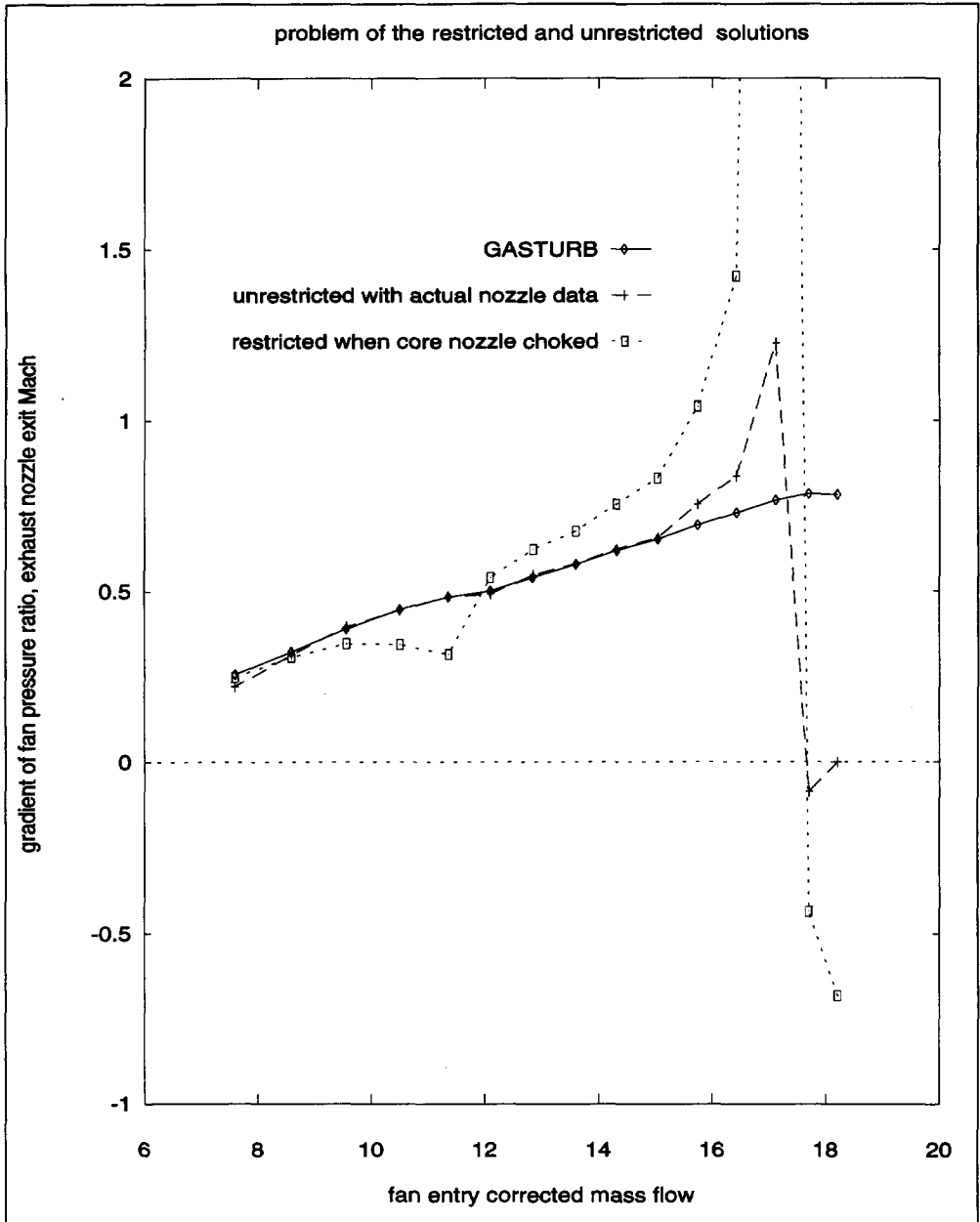


FIG. 74 Gradient of the fan pressure ratio as calculated by *GASTURB* and the closed form solutions. The original *restricted* and *unrestricted* solutions neglect the $\nabla (p_{amb}/p_{out})$ term. The dashed line indicates the original *unrestricted* solution but using actual and not the calculated core nozzle pressure gradient.

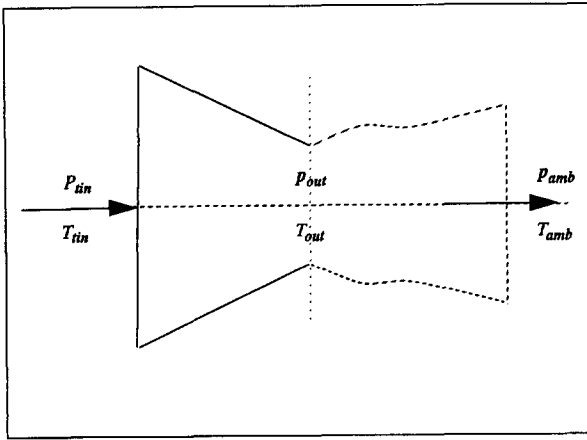


FIG. 75 Imaginary nozzle expands the incoming flow to ambient pressure and temperature.

$$\frac{\partial \pi_c}{\pi_c} = \frac{1}{1 - \frac{1}{2a_c} \frac{\tau_c}{\tau_c - 1}} \frac{\partial m_1}{m_1}, \text{ or} \quad (203)$$

$$\nabla_{m_1} \pi_c = \frac{1}{1 - \frac{1}{2a_c} \frac{\tau_c}{\tau_c - 1}}$$

We see that they are remarkably similar, except that *Cockshutt* made too many simplifying assumptions. He should have obtained the gradient with respect to the non-dimensional mass flow parameter μ_1 instead of the mass flow rate m_1 . It is a mistake because the pressure and temperature at the compressor inlet are not constant along the *ERL* of the high pressure spool. Should we be surprised at the remarkable similarity? The answer is no, since the off-design equations are the same, and one is the differential form of the other. The author expects that *Wittenberg's* solution was much more accurate than the *Cockshutt* solution because of,

1. the above mentioned unnecessary simplifying assumptions
2. not incorporating a term for variation of efficiency, i.e. the ∇a_c term
3. the accumulation of errors inherent in continuous extrapolation using gradient values

Now comparison of the above result with our closed form solution (eq. 195) reveals that although we have taken more terms into consideration, the basic form of the equations agree with each other.

So the closed form *restricted* and *unrestricted* solutions are a kind of *rectified* and *extended Cockshutt* result. The closed form solution for a *well designed well matched* spool however uses first stage turbine characteristics and thereby eliminating the need to distinguish between *restricted* and *unrestricted* spools and is not directly comparable with the *Cockshutt* or *Wittenberg* method.

To summarize, *Wittenberg's* method has the advantage that it is simple and does not suffer from accumulation of errors. The disadvantages are,

1. the result is not closed with respect to μ_1 and hence not interesting for a generalized network approach
2. the $\nabla \pi_b$ was not included (π_b varies by 2% in the test hp-spool)
3. the ∇m_{rat} term was not included. This term is significant if there is significant air bleed from the hp spool

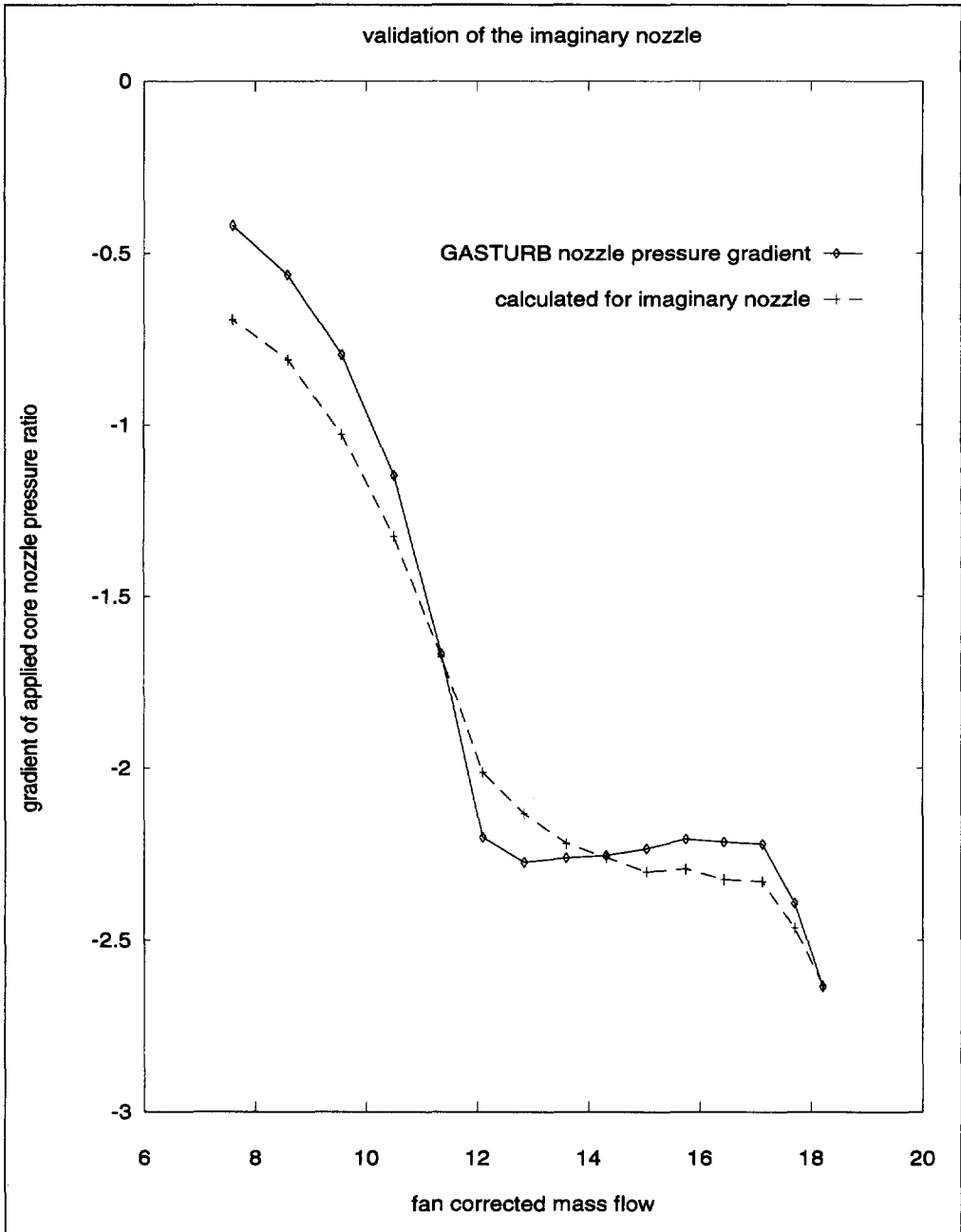


FIG. 76 Comparison of actual and calculated applied nozzle pressure gradient. The imaginary model captures the actual behaviour of the model very well.

7.2.3 The Low Pressure Spool Calculated with *Unrestricted Outlet* Solution

The original *unrestricted* solution makes use of the nozzle characteristics and neglects the $\nabla(p_{amb}/p_{out})$ term (see section 4.4.1.4). At the time this seemed a good assumption since modelling that term appeared a difficult task and since this assumption was consistent with that made in the *Wittenberg* method who reported very good results by comparison with real engine data. In addition it was anticipated that any possible problems would occur in the region where the core nozzle is choked, in which case the lp- spool could be regarded as a restricted spool which was already validated for the hp- spool. *Fig. 72* shows the consequence of neglecting the $\nabla(p_{amb}/p_{out})$ term in modelling the variation of core exhaust nozzle applied pressure ratio. The errors increase as the applied pressure ratio approaches its critical value and continues into the choked and super choked regions (moving leftwards in the figure).

Consider *Fig. 74* which plots curves of $\nabla\pi_c$ versus μ_1 for several solution. The solid line represents the actual data from *GASTURB*. The dotted line with square data points represents *restricted* and *unrestricted* solutions corresponding to choked and unchoked core nozzle respectively, where the $\nabla(p_{amb}/p_{out})$ has been neglected to derive the solutions. The “dash-dot” line with cross points represents the core nozzle exit mach number and shows when it becomes choked. It can be seen that the lp- spool clearly does not behave in the *restricted* way even when the core nozzle is choked!. Further the *unrestricted* part of the curve only gives a good result at very low mach numbers.

Now what if the lp- spool behaves *unrestricted* even when the core nozzle is choked? In order to find out, the *unrestricted* solution was applied also in the choked nozzle region, but this time actual pressure gradient of the core nozzle was used, since the assumption that $\nabla(p_{amb}/p_{out})$ is negligible is clearly wrong as shown by *Fig. 72*. The result is the dashed line with “+” data points in *Fig. 74*. It is perfectly superimposed on top of the *actual* (i.e. *GASTURB*) solid curve for the majority of the off-design range, but again begins to diverge erratically in the upper half of the “super-choked” region.

The following conclusions can be made at this stage,

1. the assumption that $\nabla(p_{amb}/p_{out})$ is negligible, is incorrect (except where core nozzle exit mach number is low) and a better model of the nozzle expansion process is required which takes this term into account
2. the lp-spool does not necessarily behave in the *restricted* way as anticipated

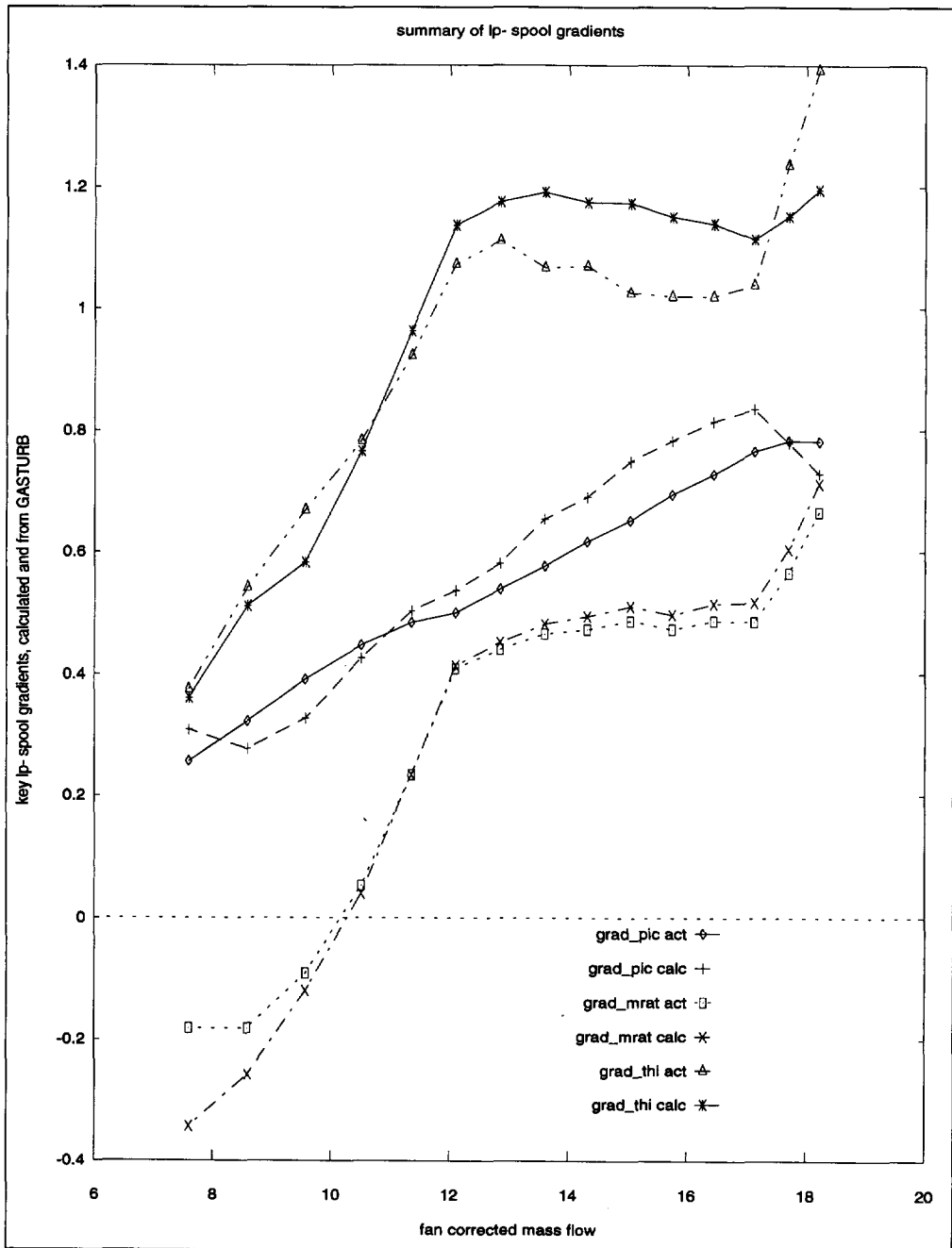


FIG. 77 Summary of calculated and GASTURB gradients for the low pressure spool.

3. the original *unrestricted* solution is basically correct provided a good model of the pressure gradient of the nozzle is available
4. the divergence in the super choked regime cannot be due to the modelling of the nozzle

Actually the reason behind the divergence is quite simple and has nothing to do with the nozzle. Looking at the *unrestricted* solution (eq. 195) we can see that the result tends to infinity if the denominator becomes zero. The denominator tends towards zero as the ratio $\tau_c / (2a_c (\tau_c - 1))$ tends to unity. The presence of C does not allow tending to infinity but cannot stop the diverging behaviour. This can occur where τ_c or π_c are relatively small, say 1.2 and 1.8 respectively, and a_c is relatively high, say 3.185 and this is close to the design point of the test engine. Low values of τ_c and π_c typically occur in high bypass turbofan engines which are included in the scope of this thesis.

So far then two matters remain unresolved, namely how to,

1. model the pressure gradient of the nozzle properly without neglecting the $\nabla (p_{amb}/p_{out})$ term
2. avoid zero's in the solution

The first item above is quite difficult, because we are expected to model the expansion outside of the actual nozzle. Furthermore, the pressure gradient must be only a function of the current operating point of the nozzle.

Consider the imaginary nozzle in Fig. 75 where the flow is expanded to ambient pressure and temperature. Fig. 73 breaks this expansion process up into several separate processes. Since the flow is expanded to T_{amb} , the velocity of the flow would not be high at the ambient station. This type of expansion may appear unrealistic at first, since an exhaust nozzle is usually modelled by a near isentropic expansion process. It is easy to see and accept that external to the nozzle, the flow will expand until its static pressure is equal to the ambient pressure. At that point the static temperature of the exhaust flow will still be much higher than the ambient temperature. However we can imagine that at some station downstream, the interaction between the exhaust gas and ambient air (e.g. mixing etc.) will be such that the static temperature of the exhaust gas will be equal to the ambient temperature. The dotted line in Fig. 75 then indicates the borders of the space that the exhaust gas occupies. Within these borders, there could be ambient air, but that is not of interest to the current study, since we are only interested in the initial and final properties of the exhaust gas.

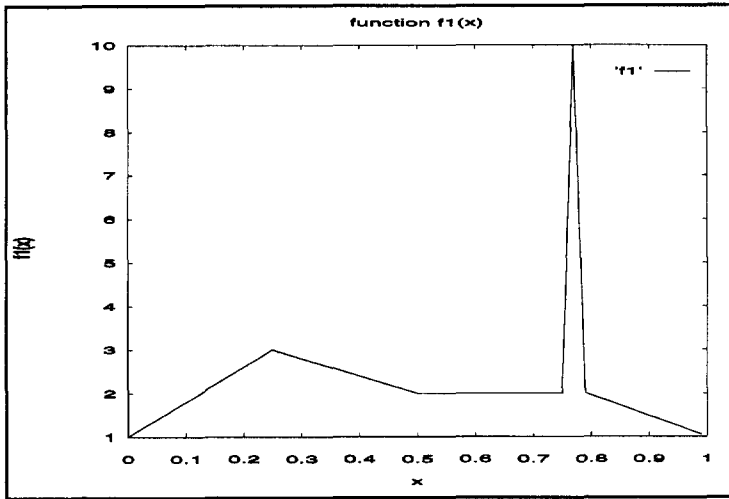


FIG. 78 a) $f_1(x)$. The 3-dimensional test merit function is the multiplication of three 1-dimensional functions $f_1(x)$, $f_2(y)$ and $f_3(z)$

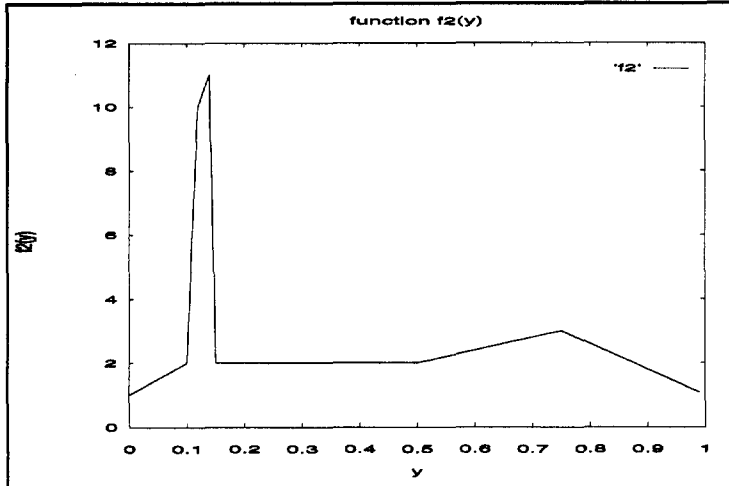


FIG. 78 b) $f_2(y)$

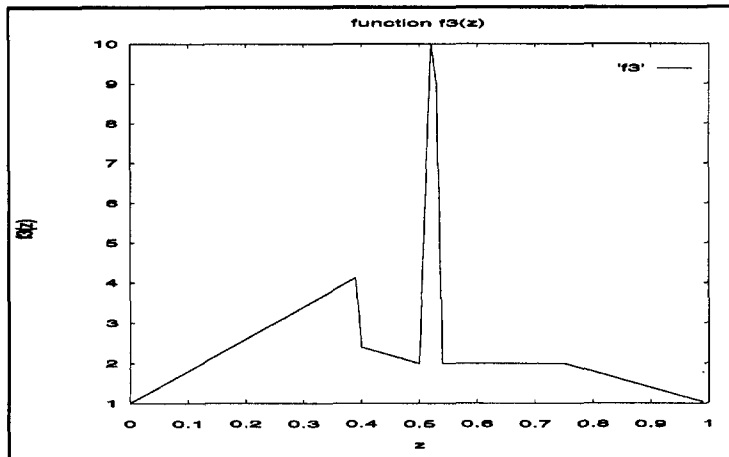


FIG. 78 c) $f_3(z)$

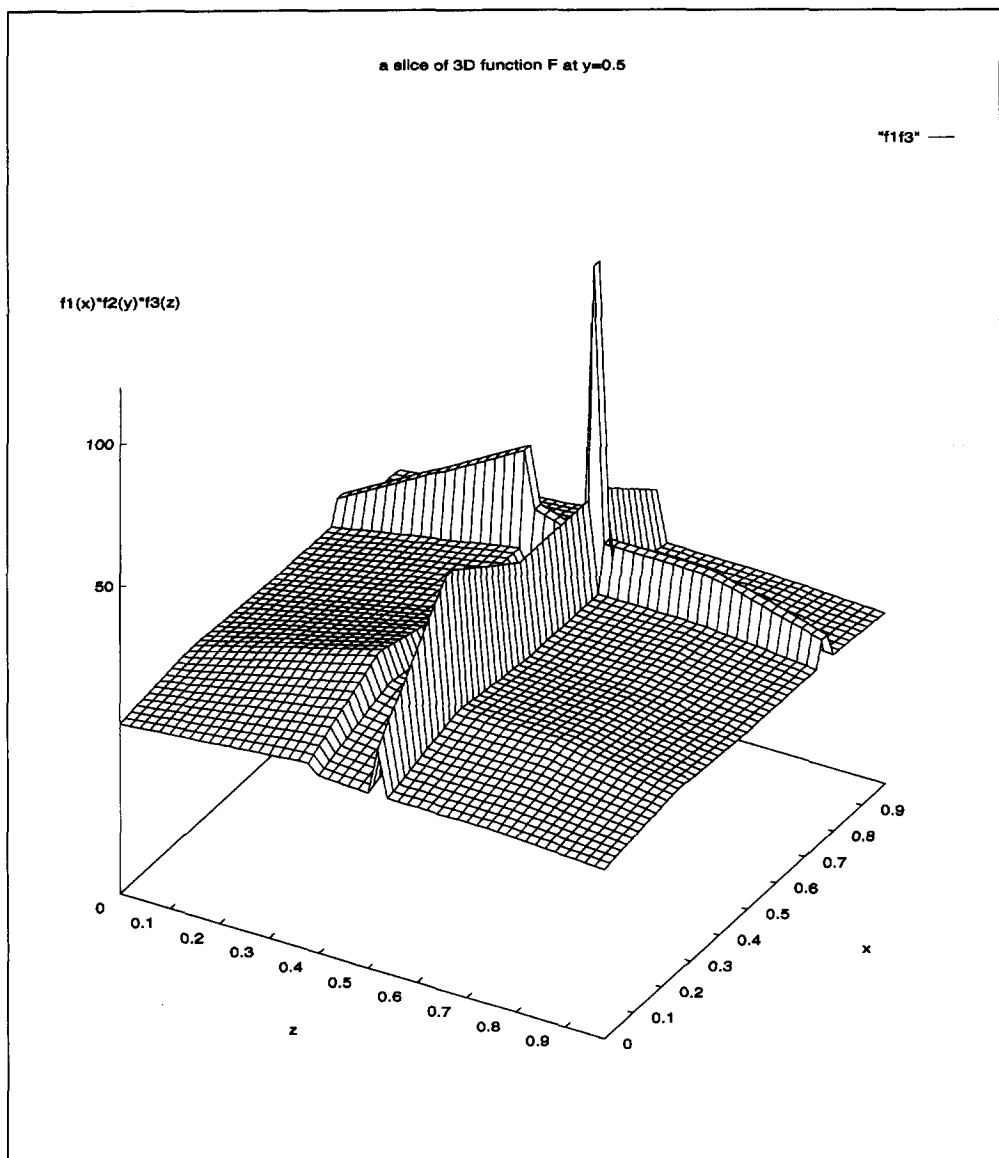


FIG. 79 A slice of the three dimensional merit function F at $y=0.5$. The basic shape of the surface remains the same at other values of y but the amplitude of the surface changes as y changes.

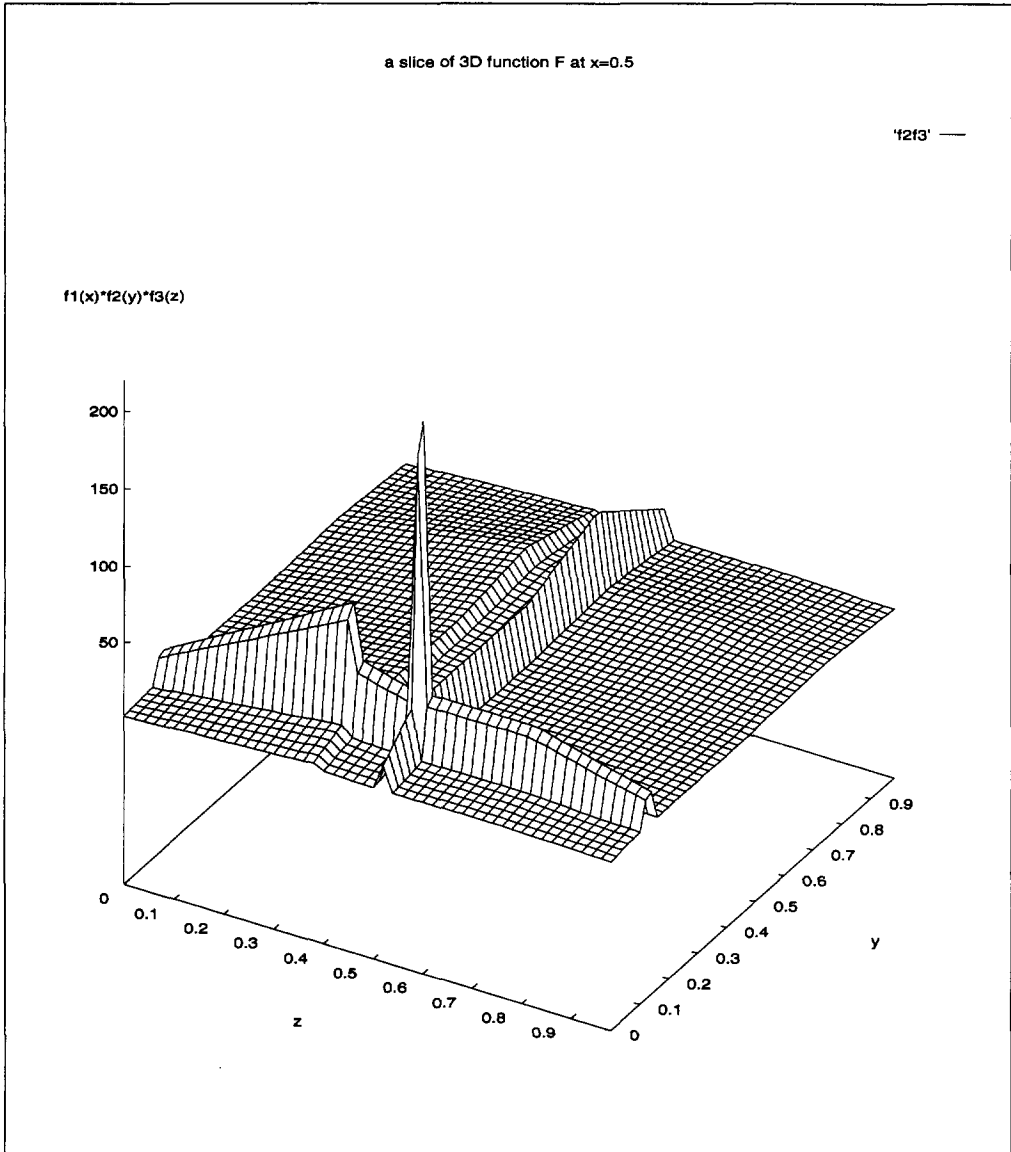


FIG. 80 A slice of the 3-dimensional function F at $x=0.5$. The basic shape of the surface remains the same at other values of x but the amplitude of the surface changes as x changes.

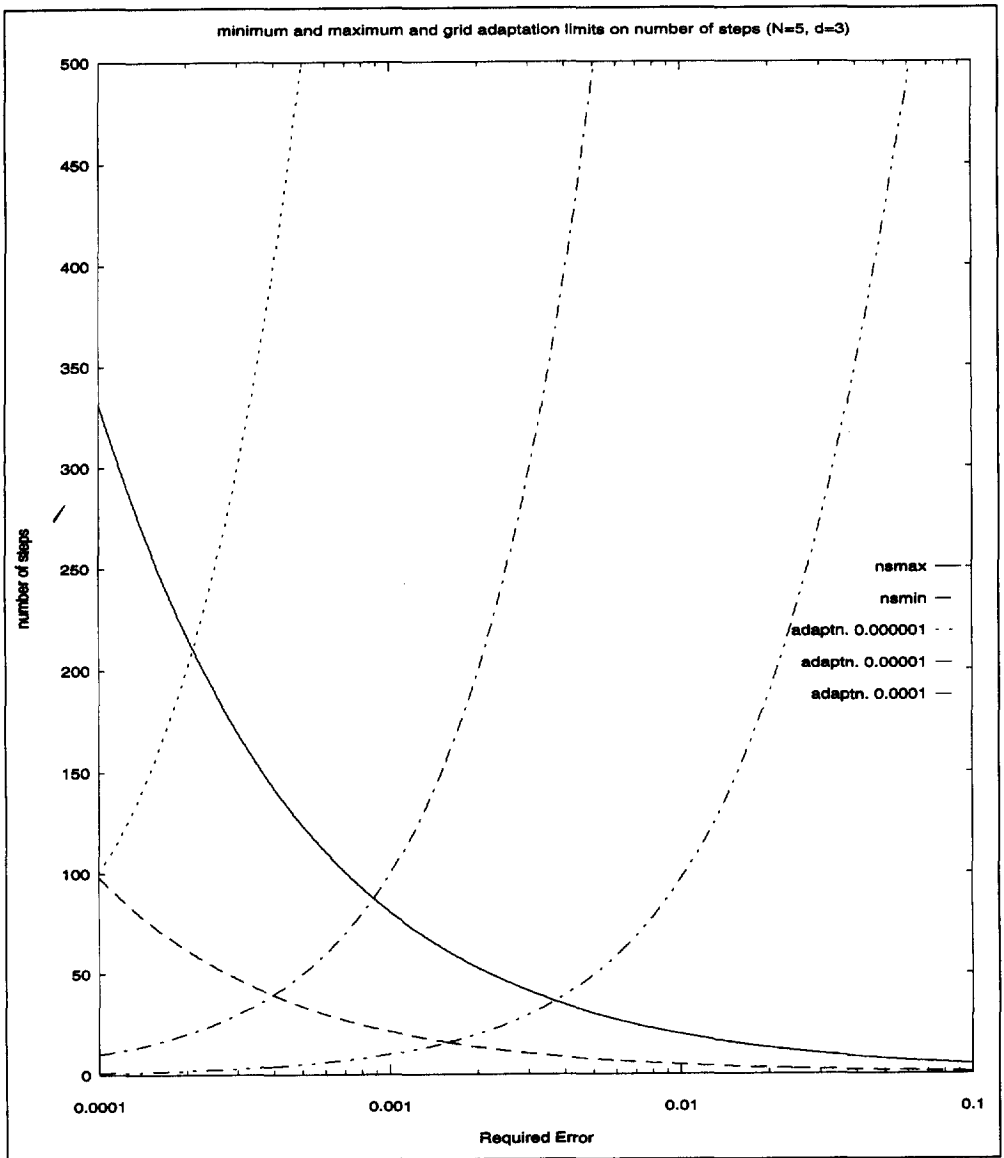


FIG. 81 Maximum and minimum number of steps and initial adaptation contours on number of steps for the original non-random adaptive grid technique (max., min. and adaptation = 0.000001:0.00001:0.0001 are represented by solid, dash, dot, dash-dot and dash-dot-dot respectively).

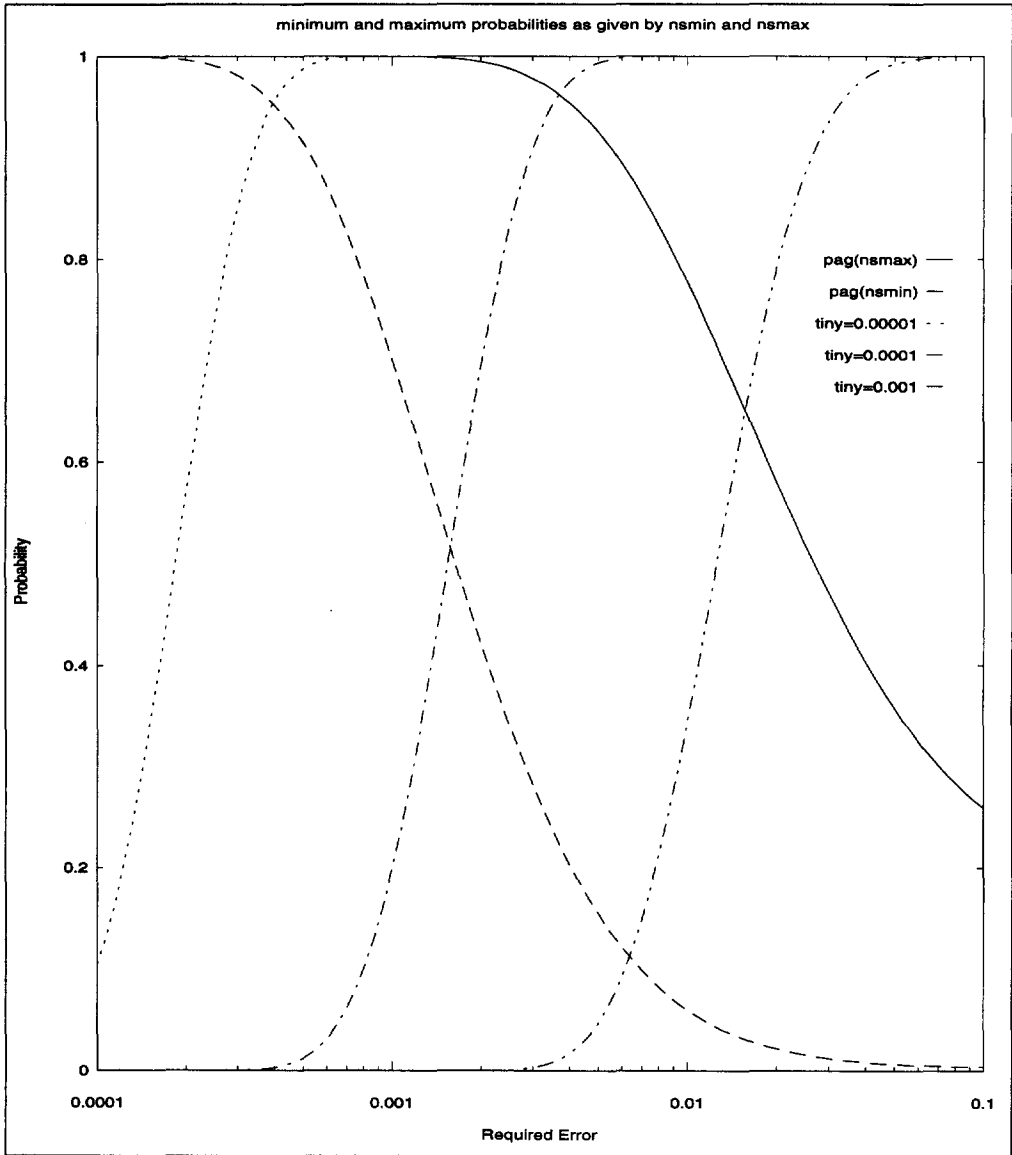


FIG. 82 Probability curves for maximum and minimum number of steps with contours of initial adaptation values ((max., min. and adaptation = 0.00001:0.0001:0.001 are represented by solid, dash, dot, dash-dot and dash-dot-dot respectively).

Treating the imaginary nozzle as a duct going through a stagnation pressure change from $p_{t_{in}}$ to p_{amb} , and applying the well known pressure loss model (e.g. same as that used for the burner),

$$\nabla_{\mu_{in}} \pi_N = \frac{2(\pi_N - 1)}{\pi_N} \quad (204)$$

where $\pi_N = p_{amb}/p_{t_{in}}$ and μ_{in} is the mass flow parameter at the inlet to the nozzle.

Transforming this result to get the gradient with respect to μ_{amb} ,

$$\nabla_{\mu_{in}} \pi_N = \nabla_{\mu_{amb}} \pi_N \left(1 + \nabla_{\mu_{in}} \left(\frac{m_{amb}}{m_{in}} \right) \right) + \nabla_{\mu_{in}} \tau_N - \nabla_{\mu_{in}} \pi_N \quad (205)$$

$$= \nabla_{\mu_{amb}} \pi_N \left(1 + \left(\frac{1}{2a} - 1 \right) \nabla_{\mu_{in}} \pi_N \right) \quad (206)$$

where $\tau_N = T_{amb}/T_{t_{in}}$ and $a = \frac{\ln(\pi_N)}{\ln(\tau_N)}$. Rearranging gives,

$$\nabla_{\mu_{amb}} \pi_N = \frac{\nabla_{\mu_{in}} \pi_N}{1 + \left(\frac{1}{2a} - 1 \right) \nabla_{\mu_{in}} \pi_N} = \frac{\frac{2(\pi_N - 1)}{\pi_N}}{1 + \left(\frac{1}{2a} - 1 \right) \frac{2(\pi_N - 1)}{\pi_N}} \quad (207)$$

Recognizing that $\mu_{amb} = \mu_1 m_{rat} \Rightarrow \nabla_{\mu_1} \mu_{amb} = \nabla_{\mu_1} \mu_1 + \nabla_{\mu_1} m_{rat} = 1 + \nabla m_{rat}$, then,

$$\nabla_{\mu_1} \pi_N = \nabla_{\mu_1} \left(p_{amb}/p_{t_{in}} \right) = \frac{\frac{2(\pi_N - 1)}{\pi_N}}{1 + \left(\frac{1}{2a} - 1 \right) \frac{2(\pi_N - 1)}{\pi_N}} (1 + \nabla m_{rat}) \quad (208)$$

Fig. 76 plots this result with solid and dashed lines representing actual and calculated nozzle gradients. As shown, the imaginary nozzle captures actual nozzle behaviour remarkably well considering that it is made up of imaginary processes. At the low power end the accuracy reduces somewhat, but this is the region where the old nozzle pressure gradient model works well.

To derive the above result the equality $\nabla \pi_N = a \nabla \tau_N$ has been used. While this is correct in the above equations for a given instant, it is not valid for any departure from a given point. Here we cannot use the correction $\nabla a \times \ln(\tau)$ as done previously for the gradient of the compressor pressure ratio because here the imaginary expansion process is very far from adiabatic or a proper thermodynamic process.

In order to get a proper correction such that,

$$\nabla \pi_N = (a + a_N) (\nabla \tau_N) \quad (209)$$

the imaginary expansion is broken down into several polytropic processes as shown Fig. 73 where the a values each represent a polytropic exponent.

It can analytically be shown that,

$$a_N = \frac{a_z}{a_x} \left(\frac{-\frac{1}{a_n} + \frac{1}{a_x}}{\frac{1}{a_x} - \frac{1}{a_y}} \right) / \left\{ \frac{1}{a_y} \left(\frac{-\frac{1}{a_n} + \frac{1}{a_x}}{\frac{1}{a_x} - \frac{1}{a_y}} \right) + \frac{a_z}{a_x} - \frac{1}{a_n} \right\} \quad (210)$$

Since all the polytropic exponents are known at any particular operating point, the exponent a_N can be calculated.

We now have enough information to form an alternative to the *unrestricted* solution. Consider the pressure equation,

$$p_{amb} \pi_c \pi_b \pi_t \pi_N = p_{amb} \quad (211)$$

applying the ∇ operator to the above and purposely neglecting the $\nabla \pi_t$ term gives our low estimate,

$$\boxed{\nabla \pi_{c_{low}} = -\nabla \pi_b - \nabla \pi_N} \quad (212)$$

similarly one can obtain (also by neglecting the turbine term),

$$\nabla \tau_b = -\nabla \tau_N - \nabla \tau_c \quad (213)$$

now the mass balance 1 to 3 in differential form can be written as,

$$\nabla \pi_c \left(1 - \frac{1}{2a_c} \right) = 1 + \nabla m_{rat} - \nabla \pi_b + \frac{\nabla \tau_b}{2} - \frac{1}{2} \nabla a_c \times \ln(\tau_c) \quad (214)$$

substituting for τ_b from Fig. 213 and the identity $\nabla \tau_c = \frac{\nabla \pi_c}{a_c} - \nabla a_c \times \ln(\tau_c)$ into the above and rearranging finally yields our high estimate,

$$\boxed{\nabla \pi_{c_{high}} = 1 + \nabla m_{rat} - \nabla \pi_b - \frac{1}{2(a + a_N)} \nabla \pi_N} \quad (215)$$

So the turbine terms were neglected giving a low and a high estimate, the final answer comes from averaging the two,

$$\boxed{\nabla \pi_c = \frac{\nabla \pi_{c_{high}} + \nabla \pi_{c_{low}}}{2}} \quad (216)$$

Fig. 77 shows the quality of the above method by plotting the calculated and *GASTURB* spool gradients as function of fan corrected mass flow. The presented solution compares reasonably well with *GASTURB*.

7.3 The Non-Random Adaptive Grid Numerical Optimization Technique

In order to convince the reader of the potential power of the non-random adaptive grid numerical optimization (See section 6.1 for theoretical derivations), a small test program was developed which applies this technique to optimization of 3-d merit functions. The author accepts that whatever merit function (F) is chosen for validation of the method, it will always be possible to say that F was a special case. A competent critic, will only be totally convinced, if a complete range of "difficult" multi-dimensional functions are successfully tested. Absolute validation is beyond the scope of this report because,

1. numerical optimization is not a central issue in this thesis
2. time limitations

In this report a particularly “difficult” 3-dimensional merit function is defined and it will be shown that

1. The non-random adaptive grid method consistently finds the true (i.e. global) optimum if the required probability P (that true optimum is found) is sufficient
2. The coupling relations derived in section 6 are valuable for setting the search parameters in practice for the desired kind of result e.g.,
 - find the location of the true optimum to within a given probability P , while minimizing the error ϵ for a given calculation effort E
 - find the location of the true optimum to within an error ϵ while maximizing the probability P that true optimum is found
 - spend the minimum calculation effort necessary to find the true optimum to within a required ϵ and P

7.3.1 The Test Function

The three dimensional test function $F(x,y,z)$ is the product of three one-dimensional functions as follows,

$$F(x,y,z) = f_1(x)f_2(y)f_3(z) \quad (217)$$

where the three functions on the RHS are shown in Fig. 78 . Each of the one-dimensional functions has two optima, one very sharp (i.e. narrow and tall) and a relatively blunt optimum (i.e. wide and short). Further the functions are very jagged (i.e. very quick or discontinuous changes in the slope of the function). Since the adaptive grid search is based on function evaluation only (i.e. not using derivative information) then the search can handle very jagged or highly non-linear functions.

To aid the visualization of F , a zx -slice of the function at $y=0.5$ is shown in Fig. 79 . The surface plot shows four optima, where the true optimum is extremely sharp. Fig. 80 shows a zy -slice taken at $x=0.5$. At other values of x , the surface has the same basic shape, except that it has a different amplitude. The location of the true optimum is defined by,

$$\begin{aligned}x &= 0.75 + s \frac{0.01}{2} \\y &= 0.1 + s0.01 \\z &= 0.5 + s \frac{0.01}{2}\end{aligned}\tag{218}$$

where $s \times 0.01$ is the width within which the sharp peak is incorporated. So s is a measure of sharpness of the peak, i.e. the lower the value of s the sharper the peak. If $s=0$ then the sharp peak is non-existent but the minimum allowable value for s should be such that,

$$s \times 0.01 > 2.0 \times \epsilon\tag{219}$$

otherwise the search is being told to find an optimum to within an ϵ where the peak does not exist in the whole of that width.

One important point to note is that in the optimum region, function $f_2(y)$ (Fig. 78) keeps increasing with y up to and including $y = 0.1 + 0.01\epsilon$. Beyond this point, the function drops rapidly (i.e. vertically) to a value of 2.0. This slope of infinity, was incorporated to make it harder for the search strategy.

7.3.2 Criteria for Convergence

As will be explained in the next section, the original non-random adaptive grid search tries to find the optimum to a given required probability to within the best error that it can. In that case, the search has been successful or converged if after the final step, the location of the found optimum is within the following range,

$$\begin{aligned}0.75 \leq x \leq 0.75 + s0.01 \\0.10 \leq y \leq 0.10 + s0.01 \\0.50 \leq z \leq 0.50 + s0.01\end{aligned}\tag{220}$$

The following averaged error parameter then gives an indication of the actual accuracy in the position of the optimum found,

$$\epsilon = \frac{|x - (0.75 + 0.005s)| + |y - (0.1 + 0.01s)| + |z - (0.5 + 0.005s)|}{3}\tag{221}$$

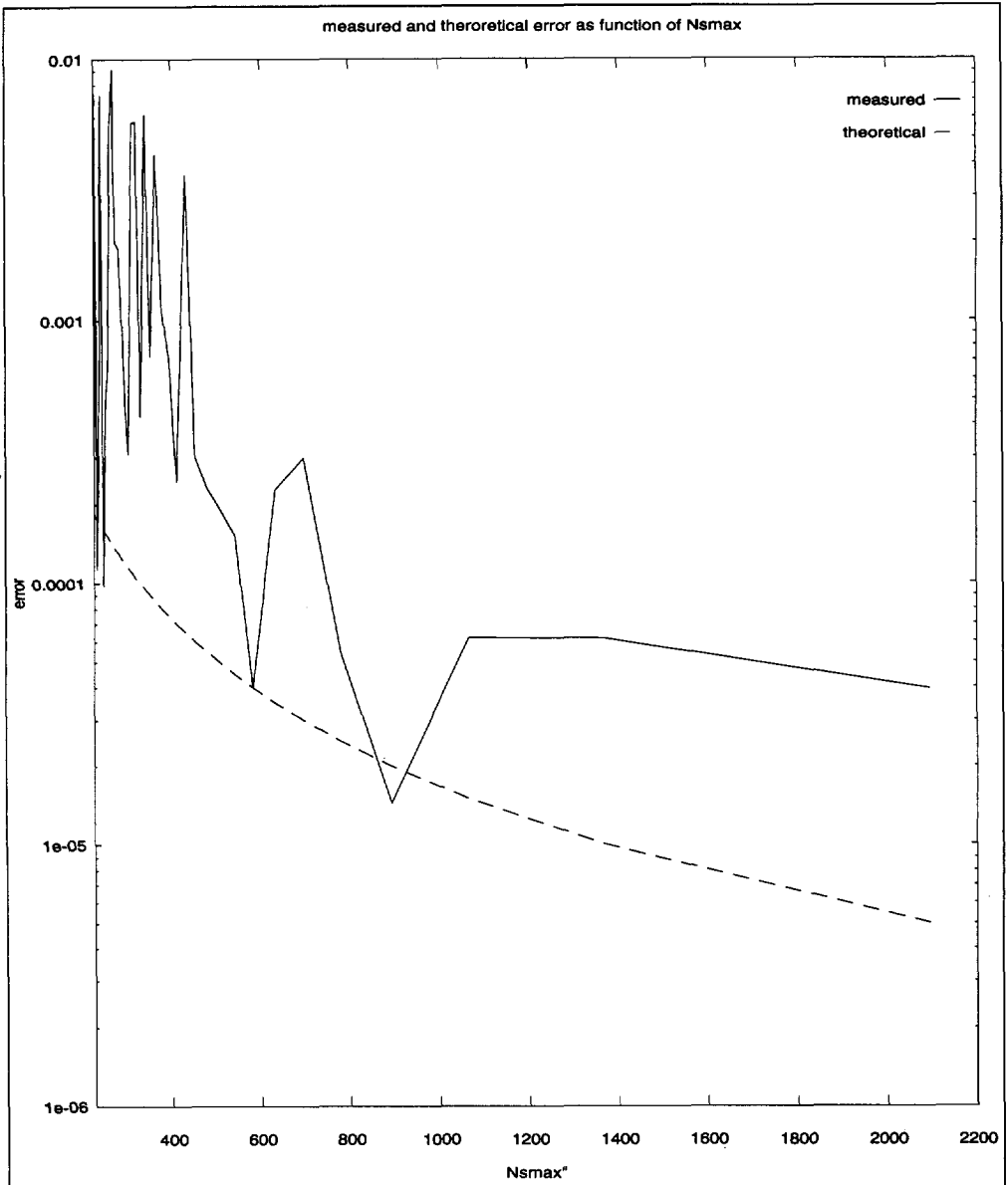


FIG. 83 comparison of theoretical error (dashed curve) and measured error (solid curve) versus maximum number of steps. (data from Table 7).

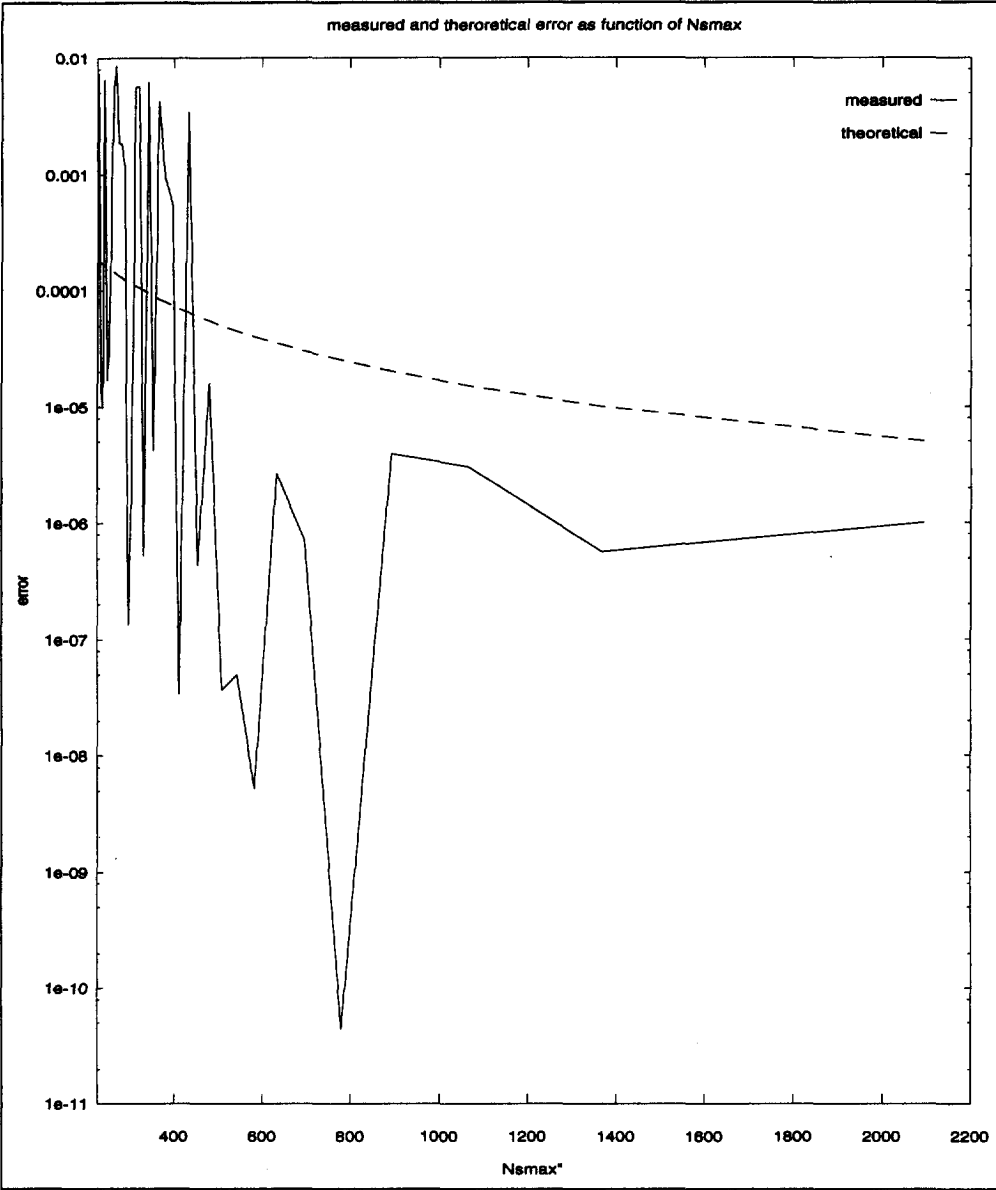


FIG. 84 comparison of theoretical error (dashed curve) and measured error (solid curve) versus maximum number of steps, where the vertical drop of $f_2(y)$ (the infinite slope) has been removed.

In the *fast search* alternative (section 7.3.4) however, the required error ϵ in the final answer is fixed and the search has converged if,

$$\begin{aligned} |x - (0.75 + 0.005s)| &\leq \epsilon \\ |y - (0.10 + 0.010s)| &\leq \epsilon \\ |z - (0.50 + 0.005s)| &\leq \epsilon \end{aligned} \quad (222)$$

7.3.3 The Original Search Method

The number of steps for the adaptive grid method has practical limits as discussed previously in section 6.1.6. $N_{s_{max}}$ is the number of steps above which the grid lines have collapsed and there is no further significant adaptation. $N_{s_{min}}$ is the number of steps below which the amount of grid adaptation in the final step is consistent with required error ϵ . $N_{s_{max}}$ and $N_{s_{min}}$ are represented in Fig. 81 by a solid and a dashed line respectively. Fig. 82 shows how these limits are reflected into constraints in the probability domain where the feasible region is the area between the solid and the dashed line. It is immediately apparent from this figure that the original adaptive grid method does not allow all combinations of ϵ and P . For example, suppose $P = 0.9$ and $\epsilon = 0.05$ is required. Fig. 81 shows that this requirement falls outside of the feasible region. So to achieve a $P = 0.9$ we are forced to accept an error of about 0.01 (intercept of $P=0.90$ and the $N_{s_{max}}$ line) which is not the value of 0.05 originally required.

Therefore more calculation effort is being spent to produce a higher quality result than actually required, implying waste of calculation effort. This wastage can be avoided as explained later in section 7.3.4.

The amount of grid adaptation in the first step is an indication of how quickly the solution is “started”. Obviously, the smaller the amount of grid adaptation, the slower the “starting”. Contours of initial adaptation are also drawn in Fig. 81 and Fig. 82. They show that in order to achieve a high level of probability, the initial grid adaptation should be much smaller than ϵ . For example if $\epsilon = 0.004$ and $P = 0.95$ are required, then from Fig. 82, the amount of grid adaptation should be smaller than 0.0001.

The above paragraphs imply that the original non-random adaptive grid search strategy can be classified as,

- find the location of optimum to a given required probability P to within the best possible ϵ .

In other words, it is not really necessary to specify a required ϵ . For a given P , the search will find the location of optimum to the best accuracy that it can. The higher the required P , the lower the ϵ . The search parameters are then as follows,

1. given P get ϵ from the P vs. ϵ curve for $N_{s_{min}}$ (the dashed curve in Fig. 82)
2. for the above value of ϵ get $N_{s_{max}}$ from Fig. 81 (the solid line)
3. set N_s for search to $N_{s_{max}}$ from previous step

Table 7 (in appendix) shows the result of the application of the above strategy to optimization of the test function F . The first and second columns show the values of ϵ and P respectively as they appear in *step 1* above. The right-most column shows the actual average error $\bar{\epsilon}$ in the results after the final step is completed. The ninth column either contains “y” or “n” indicating whether or not the solution has converged. A glance at the table reveals that the search consistently finds the optimum until $P=0.954$. Below this value, the search finds the true optimum in the majority of cases but it is not consistent any more. Further, as the required probability decreases, the average error (column 10) in the final solution increases. The data from the above table leads to some interesting conclusions about the non-random adaptive grid search. Fig. 83 shows the theoretical and measured error as function of $N_{s_{max}}$. The error decreases as number of steps increase, but for values of $N_{s_{max}}$ above approximately 1100, there is not a lot of measured improvement in accuracy. At low values of $N_{s_{max}}$, the curve is very noisy. For example, the data table shows that setting $N_{s_{max}}$ to 204, yielded a measured error of 0.00036, but increasing $N_{s_{max}}$ to 207 yielded a larger measured error of 0.0064, the opposite to what one might expect from theory. The reason for this fluctuating behaviour is that some choices of $N_{s_{max}}$ are particularly suitable for the particular merit function F and the search strategy happens to get “lucky”. However as the number of steps increase, the probability that the correct solution is found increases and getting a small error in the final answer becomes less dependent on “luck”. So increasing probability P has the effect of smoothing out the curve.

However it appears that the measured error is generally larger than that predicted by theory. The reason is the purely vertical drop (or infinite slope) of the $f_2(y)$ curve at the

location of optimum. While in practice, an infinite slope in merit function may never occur at the optimum location, the author purposely incorporated it to show that the non-random adaptive grid search still finds the optimum. although this condition causes an adverse increase in error, *Fig. 83* shows that the measured errors of order of 0.00001 are respectable.

To support the discussion in the preceding paragraph, the optimization was repeated but this time, the function $f_2(y)$ did not have a vertical drop or infinite slope at the optimum (i.e. like $f_1(x)$ and $f_3(z)$), the modified location of the y -value of the optimum became then,

$$y = 0.1 + s \frac{0.01}{2}, \text{ c.f. eq. 220} \quad (223)$$

All else being the same. *Fig. 84* displays the new situation. The behaviour of measured error is similar to the previous case (i.e. fluctuating at first and smoothing out later) but now it is below and closer to the theoretical curve, yielding much better levels of accuracy.

7.3.4 The Fast Alternative

The approach described above is a slow approach (i.e. requires large number of steps) because we are forced to accept very high levels of accuracy, if we require a high level of probability P that the true/global optimum is found. This is not a problem for low dimensions (e.g. $d=5$) but as dimensions increase, dimensionality explosion sets in and the number of steps or calculation effort becomes critical.

The preceding section explained that the costly coupling between P and ϵ was due to the grid adaptation mechanism. So is it possible to modify the method slightly, such that the required probability P and the required error ϵ can be stated separately? The answer is yes, by separating the actual number steps for the search from the number of steps used in the beam equation. Rewriting the beam equation (eq. 144),

$$I_{k_{new}} = I_{k_{old}} + J \frac{1-\epsilon(N-1)}{\pi N_{s_{beam}}} \times \left[S_1 l \sin \left(\frac{\pi}{l} I_{k_{old}} \right) + S_2 (1-l) \sin \left(\pi \left(\frac{I_{k_{old}} - l}{1-l} + 1 \right) \right) \right] \quad (224)$$

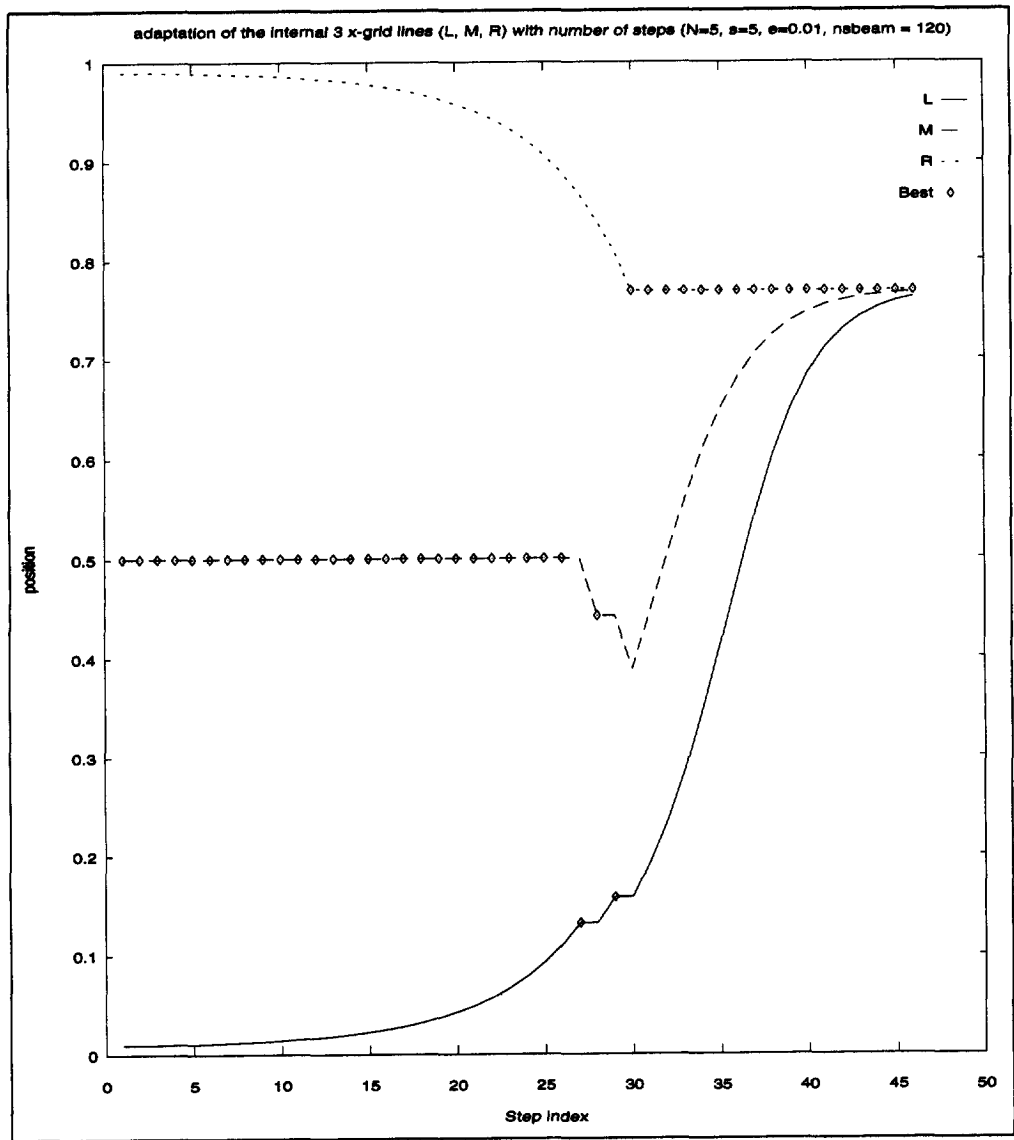


FIG. 85 Movement of the internal x-grid lines as the search progresses. The adaptation is slow in the early and final steps, but faster in the intermediate steps.

The idea is set the beam number of steps $N_{s_{beam}}$ to a relatively high value, but stop the search at a smaller number of steps, as determined by the required P and ϵ . Once a value is chosen for $N_{s_{beam}}$, the maximum number of steps $N_{s_{max}}$ cannot be calculated as before (i.e. eq. 155) because the actual number of calculation steps is not equal to $N_{s_{beam}}$.

For a given $N_{s_{beam}}$, $N_{s_{max}}$ is estimated by finding after how many steps, a grid initially positioned at $I = \epsilon$ will have moved the maximum distance towards an optimum positioned at $I = 1.0$. Substituting $l=I$, $I = \epsilon$ in eq. 224,

$$\begin{aligned}\delta_J &= J \frac{1-\epsilon (N-1)}{\pi N_{s_{beam}}} \sin(\pi I_J) \\ I_{J+1} &= I_J + \delta_J \\ J &= 1, 2, \dots, N_{s_{max}}\end{aligned}\tag{225}$$

$N_{s_{max}}$ is defined as the value of J beyond which there is no increase in I_J .

Application of the *fast scheme* involves the following steps,

1. given N and required P , ϵ
2. choose $N_{s_{beam}}$
3. calculate $N_{s_{max}}$ as explained above. After this many steps the search is to be terminated
4. the expected probability P is then calculated using $N_{s_{max}}$ and ϵ (eq. 138). If P is not high enough then increase $N_{s_{beam}}$ and repeat steps 1 to 4 until P is satisfactory
5. start the search and terminate after $N_{s_{max}}$ steps.

The above strategy was tested in practice on the test merit function F . Table 8 (in appendix) shows the results obtained for a required error of 0.01. For $N_{s_{beam}}$ values higher than 100 the table shows that convergence is consistent. Further the number

steps that was actually necessary for convergence was less than but not too far from $N_{s_{max}}$, hence the calculation of $N_{s_{max}}$ given above is realistic. Comparing the result of

the current table and Table 7, we see that the *fast search* needs 43 actual steps for consistent convergence, whereas the same figure for the *original search* is 227. Therefore the *fast search* is more than 5 time faster than the original search, in this case.

In order to show how the grids adapt as steps progress, the position of the internal 3 x -grid lines were plotted in Fig. 85 corresponding to the shaded row in Table 8. Two of the grid lines are placed at the edges (i.e. at $x=0.0$ and $x=1.0$) and cannot move. The rest of grid lines namely the left (L), middle (M) and right (R) are initially placed at $x = \epsilon$, $x = 0.5$ and $x = 1.0 - \epsilon$ respectively. The data points display the position of the best point after each step. The figure shows that until 26 steps, the best point is positioned at $x=0.5$. So the left and right grid lines adapt towards the middle, slowly in the beginning, getting faster as the steps progress. After 27 steps, the best points switches position to the left grid line, pulling the other two grid lines towards it. Once the right grid line move to about $x=0.77$, the best point remains there pulling the other two grid lines towards itself. As the steps progress further, the grid adaptation becomes slower until the grids collapse on to one another.

It is important to remember at this stage, that the grid lines are simultaneously adapting in all three x , y and z dimensions and the choice of the best point does not depend on the x -value alone.

In the original search (i.e. not the fast version of current section) the convergence of the grid lines at the end would be much slower and would occur along a much wider range of steps, hence the much higher accuracy and calculation effort associated with it.

Table 9 (in appendix) shows a much tougher case of optimization test for the fast scheme. The required error is now 0.001 and the sharpness of the true optimum has been increased ($s=2$). Here we can see that the search has more difficulty converging. However as $N_{s_{beam}}$ is increased beyond 1350 the level of P is high enough to ensure consistent convergence.

It has been explained that as dimensions increase an explosion in the calculation effort sets in. It was also discussed that if dimensions are high the lowest possible number of grid lines must be used, and this minimum for N was set to 4 grid lines, two grid lines remaining fixed in position at the edges and two internal grid lines adapting their positions. It was mentioned that, 4 grid lines would require more number of steps to find the

optimum than 5 grid lines, but the overall calculation effort would be less. Table 8 shows that five grid lines ($N=5$) requires 43 or more actual steps to consistently find the optimum. The corresponding number of steps for four grid lines ($N=4$) is 83. Using these number of steps, the calculation effort (*eq. 139*) for $N=4$ and $N=5$ are 664 ($=83(4-2)^3$) and 1161 ($=43(5-2)^3$) respectively. So $N=4$ gives more return on investment than $N=5$ and the difference grows rapidly as dimensions increase.

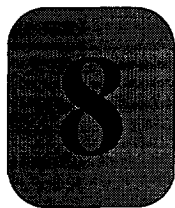
7.3.5 The Original Scheme or the Fast Scheme?

In situations where the calculation effort is an issue, the choice of whether to use the original search or the fast scheme of the preceding two sections depends on the particular application. For example, suppose a two spool turbofan cycle is to be optimized with the overall efficiency of the engine as the merit function F , with bypass ratio, fan pressure ratio and overall pressure ratio as independent variables. Now consider the following objectives for this exercise

1. Find the optimum combination of bypass, fan and overall pressure ratios to get a base-line engine for further manipulation/modification
2. Find the maximum possible overall efficiency for the engine cycle

The first objective above does not require a very high level of accuracy, but it does require a high level of probability P to ensure that the true optimum is found. In the original search, this high value of P is only possible at expense of a high accuracy and hence wastage of calculation effort. The fast technique however can deliver a high P at a lower accuracy

For the second objective above, a high level of accuracy in the values of the independent variables is essential, because only then can the maximum possible overall efficiency be accurately determined. Therefore the original search is the better choice here.



Discussion and Conclusions

To the best of the author's knowledge, none of the existing design or analysis systems have tried to comply with the natural design philosophy as the absolute highest priority. It is the belief of the author that the human designer stands a better chance of reaching a good design solution if the unnatural obstacles are eliminated from this activity. In the limit, the most "natural" design activity is that which takes place completely within the human mind without the aid of any additional equipment. In this case, it must be possible for the human to perform all design/analysis/optimization activities within his/her body. This obviously restricts the complexity of the type of object that can be designed in this way. An example of a design activity which closely approaches this limit is the sculptor creating a sculpture, which takes place without any external computational or mechanical aids.

As the complexity of the *Object* increases, so does the necessity to rely on external computational or visual aids. Given the fact that the design of complex engineering systems (e.g. a gas-turbine based engine) can never be performed by the unaided human mind, it is a worthy goal to pursue a *Natural Design Cycle (NDC)* which arranges the external processes and equipment in such a way that their hampering influence is minimized, thereby maximizing the usage of human capabilities. Therefore, the main thrust of this research was to attempt to build a prototype conceptual/preliminary design system which gains its power by mimicking the *NDC*.

In order to discourage a competent critic from branding this research as "abstract and of no practical significance", the *NDC* concepts were successfully applied to the domain of conceptual/preliminary design of gas-turbine based propulsion systems (i.e. a very complex engineering system). In order to reach this goal, several new tools and

techniques were developed and validated. The usefulness of these tools is not only restricted to the *NDC*, but to the field of engine design/analysis/simulation and computer aided engineering in general.

Section 8.2 ties the various activities of this research together within the context of *NDC*. After reading this section, it should be clear to the reader why various tools were developed and to what extent the author has been successful in obtaining a *NDC*. The author recognizes that to be fully convinced of the power of the *NDC* approach, the reader has to resort to using some imagination. The difficulty here is that the author claims that his design system has a natural feel but the *reader* can only be convinced of that if he/she also become the *user*, i.e. if he/she can sit behind the workstation and try to design an engine, and if he/she possesses some previous experience with other computer aided design/analysis systems. For those readers who are not convinced of the *NDC* approach, section 8.3. describes the contribution of this research to analysis/simulation of engines and *CAE* in general.

8.1 The Influence of Improving Computer Technology

The first appearance of the digital computer and their widespread availability had the effect of shortening lengthy calculations or procedures. However the same work could still be done manually with the hand calculator or slide rule. At that stage, the computer had no overall effect on the way engineering problems were formulated and solved, just reducing the pain of doing the same work manually. For example consider “*Synthesis of Subsonic Airplane Design*” by Torenbeek^[64] which is a standard and popular reference in its field. With the aid of the complete set of formulae contained in this book, it is possible to manually attempt conceptual/preliminary design optimization of subsonic aircraft. Programming these relations into a computer aided design system however, makes the whole process much more convenient and user friendly. This was the motivation behind the creation of *Aircraft Design & Analysis System (ADAS)*^[3] also a product of the Aircraft Design & Flight Mechanics Group, Faculty of Aerospace Engineering, Delft University of Technology.

Increasing processing speeds, machine accuracy and memory lead to more rigorous modelling of physical phenomena. For example, an aerodynamic module^[42] (i.e. a panel code) and finite element structural analysis codes were added to *ADAS*. Here there was a tendency to move away from “global” methods towards more complex iterative solution schemes. At this stage, improving computer technology was affecting the way in which problems were formulated and solved, i.e. resulted in more detailed

modelling of physical phenomena at much smaller scales. The role of the computer could now be classified as a number cruncher as well as doing the same work more conveniently.

Hence the role of the computer in engineering design evolves in order to absorb the increase in technology level. One direction in this evolution in the last decade or so, has been to give computer aided procedures a “natural feel”. For example most currently developed software employ some type of *Graphical User Interface* because it is much easier for the human to interact by *picking, moving, connecting* or *turning* (e.g. a dial) than a long series of question/answers or typing data files.

This research was primarily concerned with showing the potential power of the *NDC* when applied to the conceptual/preliminary design of gas-turbine based propulsion systems. The restriction to conceptual/preliminary design domain was necessary because a key requirement of *NDC* is speed of analysis, and the required level of detail and accuracy in this domain makes it possible to achieve high speed on a current technology workstation.

The restriction to conceptual/preliminary design domain does not take away the significance of this work to design in general because the resulting *CAGED* system provides a glimpse of a not very distant future when computers are so fast that accurate detailed calculations can be combined with *NDC*, thereby making it possible to detail design a complete engine in extremely short times.

8.2 Significance of this Research within the Context of *Natural Design*

The author is not the first to recognize nor attempt to create a natural design system. The millions of years of optimization in nature implies that nature still holds the key to the solution of many problems and that there is a wisdom in observing natural processes and events and mimicking them in so far as practical and useful. For example insects have lift devices, propulsion, control, crash protection (and cargo!) all seamlessly integrated. Except their slow speeds, the author cannot imagine an ideal aircraft with more ideal characteristics.

The problem is the identification of what makes a design system natural, which in our case came from observing what type of design activity is naturally done without the aid

of external equipment (e.g. the sculptor creating a sculpture). The resulting criteria was as follows,

- the design cycle must be allowed to follow the natural path as visualized in figure 1.
- it must be possible to regenerate a new *object* instantaneously
- The total chain of processes that occur outside of the human body must appear to the designer as instantaneous
- an aid is provided to visualize the *object* in the 3D plus time domain
- the actuators, object and properties must be observable simultaneously or in any desired order

The potential advantages of the *NDC* in so far as the author could identify were,

- short term capabilities of the human(e.g. short term memory, pattern and contrast recognition) would be utilized
- apparent equivalence of forward and reverse analysis allows implementation of whichever is most convenient
- the human sets the sense and direction of the optimization search, thereby remaining fully aware of what is going on, allowing him/her to use his/her intuition and understanding of model behaviour to guide the search

Now some of the above statements may appear trivial. For example the author realizes that in *any* design system, no matter how badly set up, if it involves the human, then his/her intuition and experience are used to guide the search for an optimum. The reader should realize that here we are concerned with how the intuition and experience can be *most effectively* utilized, not that this doesn't occur elsewhere.

The main disadvantage of the *NDC* is also related to the fact that it heavily utilizes the human intuition, experience and understanding of the model behaviour. The problem is that in practice, no matter how well the design system is set up, there is a practical limit on the number of independent design variables that the human can understand and keep track of during the optimization process or in other words there is a limit on the *dimension* of the *object*. The determination of the exact limit on dimensionality is user/object/system dependent and not readily quantifiable.

If the *object* definition cannot be reduced analytically, there is no choice but to use a hybrid type of design cycle in which an automatic optimization module (i.e. a numerical optimization routine) controls the value of some independent design variables thereby reducing the number of independent variables that the human has to directly control.

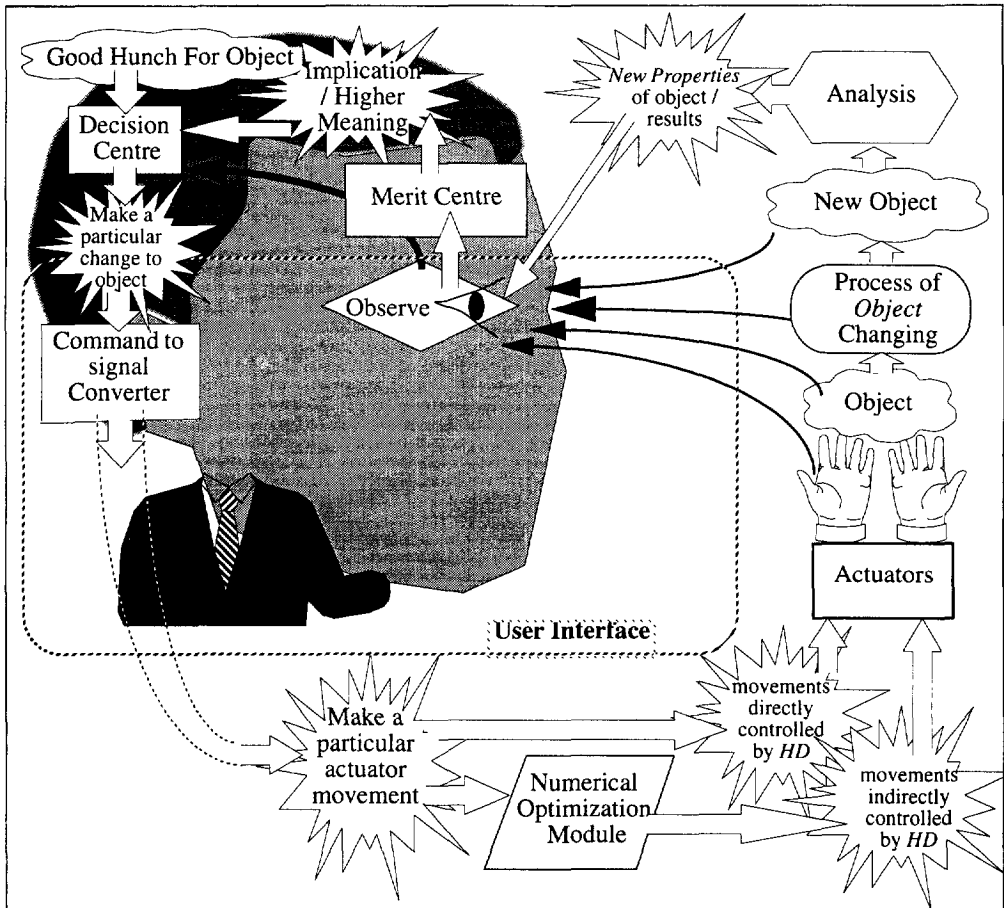


FIG. 86 The Natural Design Cycle enhanced by addition of a robust numerical optimization module. The enhancement reduces the number of independent design variables that the human directly controls making the NDC suitable for Objects whose number of independent dimensions are *too high* otherwise. The parallel arrangement of the numerical optimization module allows it to lag behind the human controlled part of the NDC thereby allowing it more calculation effort than could be afforded in a sequential setup.

Fig. 86 visualizes the NDC, enhanced with the addition of a numerical optimization routine. Here there is no significant change in the function of the designer who proceeds as before, except that this time he cannot directly change the value of some independent design variables. Instead these values are set by a numerical optimization routine.

For a simple example, suppose that the *object* is a total aircraft fitted with a two spool turbofan engine which the *HD* attempts to design/optimize. Suppose further that in relation to the engine, the following independent design variables are to be considered,

1. design point flight mach number and altitude
2. design point bypass ratio, fan pressure ratio and overall pressure ratio

Due to the large number of design variables that are related to the airframe, the designer may choose to reduce the dimensionality problem by allowing the second set of variables above to be chosen automatically by a numerical optimization routine such that the specific fuel consumption is minimized. Now every time he/she varies the values of design flight mach number and altitude, the numerical optimization routine finds the best combination of fan pressure ratio, bypass ratio and overall pressure ratio which deliver the best SFC for this condition.

Now we cannot just put any numerical optimization technique in the added module. Firstly, it must be *robust* to allow the designer a good amount of freedom in choosing the automatic design variables and merit functions, i.e. cannot use a search technique whose success depends on a particular type of merit function and independent variables. Secondly, the execution speed of the optimization module is not allowed to significantly hamper overall design cycle. In other words, the numerical optimization routine, must provide a quantitative relation between the required probability P (that the true optimum is found), the required error ϵ in the value of the automatic design variables and the calculation effort E required to achieve this. In this way it will be possible to calculate the quality of the output for a given allowable E check that this is sufficient, *prior* to setting up and starting the *NDC*.

The *Non-Random Adaptive Grid* numerical optimization method of chapter 6 satisfies the above requirements. The above discussion supplemented with that of chapter 6, justifies the heavy work involved in developing the optimization technique and its inclusion in this report. Needless to say that the setup of *Fig. 86* with the numerical optimization enhancement, has not yet been implemented in the *CAGED* system, but that this thesis provides what is required to make that step.

Based on the theoretical analysis of chapter 6, it was shown that as far as robust techniques are concerned, the *Non-Random Adaptive Grid* provides better return on investment than any robust search method in which the search volume is being reduced as the search marches, and any method in which the search points are randomly generated. Further it was shown that a particularly deceiving 3d merit function, possessing multiple optima of various degrees of sharpness and semi-infinite gradient

at the location of optimum in the y -dimension poised no threat. At this stage, this serves as a mere *good* example and full validation would of course come from testing the technique on a full set of standard difficult merit functions and checking to see whether the analytically derived coupling relations between E , P and ϵ are realistic.

As discussed previously, one requirement of *NDC* is that the designer must be given the opportunity to generate any object and in the ideal limit instantaneously. In the domain of engine design, this translates to the capability of defining any engine configuration. This was the most important reason behind developing the graphical user interface (*GUI*) which makes it possible to define a desired engine configuration by selecting from a standard set of components and visually connecting them into a network of forward flowing information. In the *CAGED* system, the definition of an engine configuration typically takes a few minutes at a relaxed pace, if the user starts from scratch. The capability of recursive inclusion of pre-defined models (i.e. including a model within a model within a model and so on) reduces the configuration definition effort even further. The definition of an engine model in general requires definition of engine configuration and assigning values to the engine model parameters (e.g. setting bypass ratio to 5). It was possible to make the latter procedure “instantaneous” by coupling the values of the independent variables of the engine, to physical dials (supplied with Silicon Graphics Workstations) so that the user can instantaneously change the value of a particular variable by turning the corresponding dial. If the engine configuration definition phase was also instantaneous, then *CAGED* system would be significantly closer to the *NDC* ideal, but this was not possible.

In the *CAGED* system, the *GUI* also serves as an overlay to all system functions. It was possible to minimize the computational overhead by developing it from scratch using no external systems or routines other than the *IRIX Graphics Library* that is supplied with the *IRIX* operating system of *Silicon Graphics* workstations.

Once an engine configuration is visually defined, the user proceeds by defining the independent and dependent variables of interest using pop-up menus and/or typing in the necessary mathematical relations. Five standard dependent variables have been defined since they are often required, namely specific thrust, specific fuel consumption, overall efficiency, propulsive efficiency and thermal efficiency. Two standard independent variables have also been defined, namely flight Mach number and altitude. Definition of user defined variables (independent or dependent) can be as simple as visually selecting a thermodynamic variable or property or a complicated mathematical relation as permitted by the *IRIX C Compiler*, because all these relations will be included in the generated source code for the engine model which will be compiled and linked to the *DynCarp* or *DynHist* programs at a later stage. In practice then the user has

access to all the key words included in the C syntax and to all the mathematical functions declared in the header file *<math.h>* (e.g. *sin(x)*, *cos(x)*, *fabs(x)* etc.) which in unix systems resides in the directory *"/usr/include"*. For example an independent variable could be the fan pressure ratio or fan polytropic efficiency which is simply selected visually by clicking the mouse button. Now suppose that the design point fan efficiency is to be related to the value of fan pressure ratio. In this case first the fan pressure ratio is visually selected as an independent variable, followed by typing the mathematical relationship stating the fan efficiency in terms of the fan pressure ratio. If this relation contains a syntax error, then the *C Compiler* will produce error messages pointing to the relations responsible when the source code is being compiled.

So for the particular *user-defined* engine configuration, dependent and independent variables of interest to the current design activity, the *CAGED* system generates an *ANSI C* compatible source code for on-design thermodynamic design/analysis, which is subsequently compiled and linked to the *DynCarp* and *DynHist* programs. The techniques that made the high execution speed of these engine models (discussed in detail in previous chapters) are briefly due to,

- the structure of engine models
 - in source code format (i.e. file *"gt.c"*)
 - separation of all initialization procedures (the function *"init_model()"*) from run time procedures (the function *"gt()"*)
 - including only the user-defined dependent and independent variables in the engine models rather than including a very large super-set of these
- gas properties generated at high speed from polynomials or from chemical equilibrium data (using the *Modified Newton-Raphson*) rather than performing lengthy chemical equilibrium calculations every time gas properties are required
- the quickest possible calculation route for marching through the engine network determined prior to generating the engine models (the *sequential network logic* story) avoids run time wastage of calculation effort
- analytical short-cuts/tricks to avoid lengthy iterative procedures (e.g. the closed form non-iterative solution for off-design steady state and transient performance of a generic spool)
- paying some attention to speed in developing the software (a heavy effort in this direction can result in further improvements in speed but the gain will not be of order/s of magnitude)

Two computer programs were developed, namely *DynCarp* and *DynHist* in order to exploit the high speed on-design engine models and to set up a high speed design optimization loop that mimics the *NDC* as visualized in *Fig. 1*. They were first demonstrated to the *CAD* community at the *1992 International CAD Conference* at Delft University of Technology. These programs generated some excitement as they presented a fresh alternative to numerical design optimization, which is very inflexible and leaves the user largely unaware of what is going on in the search. Numerical optimization techniques transform a physically formulated object into a numerical domain which is very difficult to fathom by human beings in general, thereby reducing the utilization of the human capabilities in the design optimization process. These two programs have been discussed in more detail in preceding chapters, but a short mention of their current features will be included in this section.

The *DynCarp* program consists of two windows on the monitor. The independent variables window and the carpet window. The former displays the values of the independent variables in the form of histograms, with arrows to indicate the direction of change in these values. The latter window displays a carpet plot of two independent variables as parameters of the carpet and two dependent variables on the *X* and *Y* axis. In order to produce a depth effect, the *Z*-axis also represents the first independent variable. The dependent and independent variables of the model have already been defined in the model definition stage explained above and the choice of which variables to display in the carpet is made visually. Now the independent values are coupled to dials. Turning clockwise and anti-clockwise increases and decreases the corresponding variable respectively. The carpet is drawn by joining 25 points, where each point is the result of a single call of the engine model (i.e. the function "*gt()*"). If the value of an independent variable is changed by turning a dial, the resulting new carpet plot appears instantaneously on the screen so that the user can directly observe the result of his variation in *real time*. The current version includes allows the following commands via nested pop-up menus,

- Set Dials Scale: the sensitivity of independent variables to dial movement can be reset
- Set Dependents Range: vary maximum and minimum values of the *X* and *Y* axes by turning the corresponding dial, and observing the result in real time
- Set Independents Range: vary maximum and minimum values of the independents in the carpet, and observing the result in real time
- Get/Set Memory: allows to store or recall a particular position of the carpet
- Get/Set Nominal: allows to store or recall a nominal position of the carpet

- Browse: allows real time rotation about or shift along X, Y, Z axis with in/out zoom
- Select Independents/Dependents: allows quick visual selection of what is to be displayed in the carpet
- Infeasible points chopped: the points of the carpet that are infeasible are not included in the drawing of the carpet. The function "*gt()*" sets a flag if a calculated point is infeasible which is later used to decide which points to include in the drawing of the carpet

These features make the program very versatile in that a best view can be selected, zoomed in or out and this best view can be changed at will as required. The *DynHist* program represents the same basic idea. Here there are three windows, containing the independent, dependent and the sensitivity histograms. Again an upward and a downward arrow represent increase or decrease in the corresponding variable. The arrows belonging to a variable become a square if there is no significant change in its value. The sensitivity histograms represent the sensitivity of a each dependent variable to each independent variable. For example one histogram represents the percent change in *SFC* with respect to one percent change in fan pressure ratio. The independent variables are coupled to dials as before and the user can vary the value of say the fan pressure ratio and see the resulting change in the dependents and all the sensitivity histograms in real time. If the number of independent and dependent histograms are m and n respectively, then there are $m \times n$ sensitivity histograms to compute every time an independent value is changed. This means that $m \times n$ calls have to be made to the engine model, prior to updating the display. Since the number of effects is large, e.g. arrows changing direction, histograms moving to new positions etc., it is at first a little difficult to use this program for optimization. But once the user gets some hands on experience, he/she quickly amasses experience and understanding about the engine model behaviour which aids in finding an optimum.

Only when the development of the following was complete,

1. The *GUI* enabling visual definition of the engine
2. The generation of high speed on-design engine models for user defined configurations and variables
3. The *DynCarp* and the *DynHist* programs

could the claim be made that an engine conceptual/preliminary design system was created that closely mimics the *NDC* of figure 1, in the on-design thermodynamics domain. Probably the biggest conceptual challenge in the above effort was the simultaneous requirements of *high accuracy* (~99.9%), *high execution speed*

(~milliseconds) and *general configurations* from the engine models. Had we only been interested in showing the potential of a design system that mimics the *NDC* and what such a system would be like in practice, then we might have been satisfied with terminating the project at that stage.

However in practice, selection of an engine thermodynamic cycle, does not only rest in its on-design performance, but also on its performance throughout a tentative mission/envelope. The dimensions and weight of the components and of the overall engine are a strong function of off-design performance because the components are sized for the most critical conditions reached within the mission/flight envelope which is not necessarily occurring at the design point. So in order to increase the practical usefulness of the design system, research continued on how to calculate the steady state and transient off-design performance of engines to a high accuracy *and* at high execution speed. Now all the necessary technique was already developed to calculate any network in which the flow of information is always downstream (i.e. same in the on-design case). Unfortunately, in classical off-design analysis, there is a severe backward flow of information which comes from matching the compressor and the turbine. This problem gets worse as there are more than one spool, where a spool is included within a spool within a spool and so on. So the concentration was focused on how to eliminate this unwanted backward flow of information such that the off-design steady state of engines could also be solved as a network with information flowing downstream. A generic spool with bypass was drawn up (section 4.4) and differential analysis (i.e. the ∇ operator, a type of dimensionless log derivative) was applied to the off-design equations of this spool. Further analytical manipulation lead to the following three categories of solutions,

1. Spool with restricted outlet: applicable to high pressure and intermediate pressure spools
2. Spool with non-restricted outlet and pressure recovery: applicable to low pressure spool
3. Well designed and well matched spools: applicable to all spools regardless of type restriction at outlet, requires minimal information about the turbine stage which comes from the velocity triangles at design point.

The above solutions are in *closed form* and direct (i.e. *non-iterative*) which require very little amount of information, ideal in the conceptual/preliminary design stage where detail information is scarce. All the above three solutions use the value of the spool parameters at the current steady state point, and yield the rate of change of these parameters with respect to the corrected mass flow parameter at inlet to the spool. In the first solution above, turbine parameters vanish and the off-design steady state becomes purely a function of the compressor and the burner. The second solution above, turbine

parameters and downstream expanding nozzle parameters enter the arena, making the equations somewhat more complicated, but not changing their *closed form* nature. The third solution requires extra information about the design point velocity triangles of first turbine stage (impulse, symmetric or zero swirl etc.). The solutions do not require component maps, but do require information about how compressor or turbine efficiencies vary along the *ERL*. Now if accurate compressor and turbine efficiency information is available, then they can be used. However all is not lost if this information is not available. The following simple observation helps in this regard,

- in most engines, the compressors are well designed and well matched. In other words the *ERL* closely follows the *Backbone* of the compressor and almost always crosses it at one point
- the backbone is the locus of maximum efficiency (for a given shaft speed) points on a compressor map. Since the efficiency curves are very flat in the proximity of the backbone, and since the *ERL* is quite close to the backbone, then the variation of compressor efficiency on the *ERL* is quite accurately modelled by the backbone efficiency.

Johnsen & Bullock^[29] give a simple method for determination of backbone efficiency which is based on the actual performance of several transonic and subsonic compressors. A less accurate but more convenient method is given by *Flack*^[18] which gives a mathematical model of characteristics of compressors, turbines etc.

The solutions make it possible to solve the off-design steady state of multiple spools by starting from the innermost spool and working outwards using the values of spool parameters at the current steady state point. Once these closed form solutions were validatedⁱ (99% accuracy at the design point falling to 96% at part load) then the unwanted backward flow of information in calculation of the off-design performance was successfully eliminated. The structure of the off-design engine model then became nearly identical to the on-design engine model, except the addition of a module, to update some key spool parameters (e.g. $\pi_{c,\varnothing}$ for each spool) prior to calculating the network and a module to update the spool gradients after the network had been calculated. Due to lack of time it was not possible to implement this within the *CAGED* system, but since,

1. Closed form solutions for off-design were validated
2. Off-design engine model is almost identical to On-design

i. the Well Designed Well Matched solution has not yet been validated due to lack of time

3. On-design engine model is validated

then we can safely say that it is certain that the off-design engine models are *valid* and *high speed* when they are implemented in the *CAGED* system.

Acceleration of an engine, from part load to full load can also be a parameter in design, which in conceptual/preliminary design exercise is usually solved by the quasi-steady method involving manipulation of component maps in order to calculate the shaft torque at a given transient point. An analytical attempt was made to increase the speed of transient calculation by avoiding component map manipulation. Then the realization came that all the information necessary to calculate the natural transient response of a spool was already contained in the *ERL* which was already calculated in the off-design engine models. The proposed technique uses the relative position of the current transient operating point and the *ERL* to calculate the new position a small time step later, that the spool is naturally tended to go to. Unfortunately there was no time to validate this theory.

By this time, the theoretical basis was complete for attempting to build a full preliminary design system for gas-turbine based propulsion systems which complied with *NDC philosophy*. A full *PD* system places the following set of independents in the control of the *HD*,

1. network configuration of the engine
2. on-design thermodynamic variables of the above engine network, e.g. fan pressure ratio, bypass ratio, turbine entry temperature etc.
3. tentative flight envelope/mission, i.e. pairs of altitude and flight mach numbers, with one particular pair designated as the design point
4. required thrust at the design point
5. mechanical design variables and material properties of components, e.g. for a compressor, the hub/tip ratio, allowable stresses etc.

and would generate the following output (i.e. dependents) for the particular situation as depicted by the above set,

1. on-design thermodynamic performance maps of the engine model
2. performance at all points in the mission/flight envelope
3. critical conditions reached for each component during mission/flight envelope, e.g. maximum temperatures, shaft speeds etc.
4. component dimensions that can bear the above critical conditions
5. engine annulus drawing showing assembly of components
6. weight & cost

The technique for going from the critical conditions for each component to component dimensions and weight is given by Boeing^[46] WATE code with later extension of the method to calculate costs^[47] also.

The above PD system can be classified as an *Integrated Thermo-Mechanical Design System*. Of the two approaches discussed in chapter 5, going from engine thermodynamics to engine geometry was a more practical approach to use and to implement on the computer. Figure 28 on page 101 illustrates the flow and order of various procedures in this approach. The procedure is briefly,

- depending on the value of the design point thermodynamic variables, and the required thrust, the design point engine mass flow rate is determined. The engine is then flown through the stated mission/flight envelope yielding the critical conditions for each component. These combined with the allowable stresses and mechanical design variables yields the component geometry, weight and cost.

It should be recognized that all the necessary techniques and modules required to implement the integrated approach are either contained in the results of current research or included in WATE.

Since all the procedures in Fig. 28 are high speed, then it is easy to accept that overall design cycle satisfies the high speed required for the NDC. When implemented, this system would allow the user to vary any of the independents and observe the resulting consequences in the dependents (including engine geometry and mechanical layout) instantaneously (i.e. faster than movie rate 1/24 seconds). Now any engine preliminary design activity would involve similar sort of procedures as in the proposed *Integrated Thermo-Mechanical Design System*. However in the computer aided engine PD systems that the author is aware of, one cycle of the design optimization loop takes roughly 0.5 to 2 hours which is at a serious disadvantage when compared with what this thesis claims to be possible.

8.3 Significance of this Research Out of the Context of NDC

The usefulness of this thesis is not restricted only to the NDC. This section is included for readers who are not particularly interested in the NDC approach and discusses the various tools and techniques which are of interest to CAE and engine design/analysis/simulation in general.

8.3.1 The Graphic User Interface (*GUI*)

The majority of modern applications in CAE employ some form of *GUI*. It is broadly accepted that this greatly improves the efficiency of user interaction and majority of users seem to prefer it. The *GUI* is very pleasant to use but not particularly interesting to read about unless the reader is also interested in developing one him/herself. For very large systems/projects this development is better left in the hands of a specialist in the field. At the moment, there are several development tool kits available which considerably take the pain out of designing a *GUI* which are suitable for majority of cases. However if there is a particular overriding requirement/priority that is not provided, then the software developer may obtain better results by going it alone and writing the drawing/interaction routines him/herself. In *CAGED* the *GUI* had to perform only a relatively few simple tasks if compared to large *CAD* packets, namely,

1. overlay to all system functions, i.e. there is a button associated to each function, so that pressing it calls the function, e.g. to start *DynCarp* the user clicks with the mouse in the screen button that says *DynCarp*
2. definition of an engine network of standard components
3. definition of interaction between these components, e.g. aerodynamic link, power link, heat transfer, or user defined artificial relations called *EARS*.
4. using the graphic picture of the engine to get/set component parameters

But the above had to be performed extremely efficiently, i.e. minimum memory and CPU usage, because the *DynCarp* and *DynHist* programs would be running while the *GUI* was running and any time savings here translated to longer allowable execution time for the engine models.

Each function and each engine component was assigned to a screen button. A screen button is a small rectangular RGB image with its name written under it. A total of 64 buttons were necessary which were arranged into 15 menus, each menu drawn into a separate window. The menus were all nested under a *master menu*. There are only five windows visible at any one time so they are all placed on top of each other. Each screen button has associated with it 5 windows, the one that the button itself is drawn into and 4 other. All the buttons have the same size so it is very quick to determine which button has been clicked. Clicking a screen button automatically pops up the other 4 windows associated with it if they are not in view already. For example, there is screen button called *network* in the *model definition* window, and the user has to click it if he/she wants to define the engine network from scratch. Clicking this button pops the following three windows

1. the *components window*: a set of buttons which represent the standard gas turbine components

2. the *interactions window*: a set of buttons representing the type of interactions possible between components
3. the *display window*: in which the engine network definition appears

Supposing the user wants to define a compressor feeding a burner. He/she first picks each of the two components from the *components window* and places them at a position of his/her choice in the *display window*. Then he/she clicks the two components in the *display window* and presses the *aerodynamic link* button in the *interactions window* and draws the connection. Since the *GUI* reacts instantaneously to each click of the mouse button, the process of engine definition is very quick.

A full description of all the *GUI* features and how they were attained is not necessary here, but there is one particular capability which is particularly interesting and that is the *modelling commands*. In order to make engine network definition a much simpler task, it was necessary to incorporate recursive inclusion of models, so that a model could be included within a model within a model..... and so on. Now an engine model has a lot of information associated with it, the components and station numbers, the mass, work or heat transfer between the components, the values of the thermodynamic variables etc. Normally all this model information would need to be saved in a formatted file, which would have to be read and translated back every time it is included within another model. This type of inclusion also requires some computation because the station numbers of the included model have to be renumbered, all the aerodynamic and mechanical links updated etc. Now this procedure can be made extremely simple, by realizing that all the model details (the network configuration, the values of the thermodynamic variables etc.) are a direct result of the *user interaction* with the *GUI*, through clicking the mouse buttons or the keyboard. Therefore the model can be completely regenerated by saving the *user interactions* that define each model, and feeding it back into the event queue of *GUI* every time that model is to be included. The system then reacts exactly the same as it did when the model was originally defined. Model files in *CAGED* are then extremely simple, they contain *x* and *y* positions of the mouse corresponding to each down click of the mouse buttons and the identity of each keyboard button that was clicked. This way of handling models is useful to any *CAE* system in which the *cause* is simpler or quicker to handle than the *effect*.

In conclusion, the *GUI* module of the *CAGED* system is very compact, efficient, and is applicable to any analysis/simulation exercise which requires the *Systems Approach*, i.e. building up a complex object by defining a network of components and their interactions.

8.3.2 The non-random adaptive grid numerical optimization technique

One of the products of the current research program was a new robust numerical optimization technique. This technique is an alternative to other robust optimization techniques. It was theoretically shown in chapter 6 that it gives much *better* return on investment than any *non-gradient guided* search in which,

1. the total search volume reduces in general as the search progresses
2. there exists some randomness in the generation of search points

Now since the majority (if not all) of *NGG* techniques employ one or both of the above steps, then the *Non-Random Adaptive Grid Search* proposed herein should be taken very seriously. Although the author and the reviewers of the paper^[57] are not aware of another technique that is significantly similar, it is of course quite possible that the proposed search method has either been tried before or fits into some well known category of techniques which are similar in essence. However it is unlikely that a quantitative relationship/coupling between *Probability*, *Calculation Effort* and *Accuracy* (i.e. between P , E & ϵ), has been developed elsewhere which is valid for a general *Adaptive Grid* method in which the total *search volume* may be *reducing* and/or there is a *degree of randomness* in the generation of the search points. Probably the role of the numerical optimization routine (and high speed execution capability) has never been so critical as it was in the proposed *Natural Design Cycle* of Fig. 86. In other words, there may never have been a very strong motivation to measure the success of the optimization technique, *prior* to the commencing of the search, hence the unlikely appearance of these coupling relations elsewhere.

The *coupling* relations above should not be confused with what is normally referred to as *convergence* equations. The latter gives the number of steps for the solution to converge to an optimum which is a simple function of contraction ratio or the rate of reduction of search volume. A convergence equation does not in general give a probability that the found optimum is a true optimum.

The successful optimization of a multiple-optima and difficult 3D test merit function provides sufficient grounds for further detailed investigation of the *Non-Random Adaptive Grid Search* technique to properly determine and validate the domain of its application, and to check that the quantitative coupling relations are actually realistic in this domain.

8.3.3 The *DynCarp* and *DynHist* modules

These are two programs which allow human in the loop optimization. They also serve as a powerful and convenient parameter variation technique by allowing the user to make a very large number of variations and observe and grasp the consequences in a very short time.

Their usefulness is not restricted to engine modelling, and they can be compiled and linked to any model file which has a quick execution time. For example, a high speed model of aircraft performance (e.g. a collection of relations from *Torenbeek*^[64]) could also be compiled and linked to *DynCarp* or *DynHist* as long as it complies to a few variable definition conventions.

Using high speed dynamic models is much more flexible than animation of static data. In a static data file, the range of independent variables are fixed. If during the parameter variation procedure, the user becomes aware that he/she is actually more interested in a particular sub-space of the data table, or in a space that falls outside of the data table, then the data has to be regenerated. A high speed dynamic model, allows the user to change the ranges of the independent values, on-line and in real time, which is obviously superior to animation of static data files.

Despite the above arguments, both *DynCarp* and *DynHist* programs have the added capability of animating static data files, because

1. it may not be possible to set-up a high speed model for accuracy or other reasons
2. the user may prefer to analyse data output of a sluggish but accurate program which he/she has grown to trust

Both *DynCarp* and *DynHist* can read and animate unformatted data files in ASCII text with a simple header, including the name of the variables and the number and order of independent variables in the table.

The above requires fast function generationⁱ from data and the “standard” generation techniques (for example those given in *Numerical Recipes in C*^[52]) are not sufficiently fast/efficient for animation. As explained in more detail in section 6.6 on page 152, the calculation burden is associated with,

1. locating the smallest multidimensional grid surrounding the required point

i. also referred to as multidimensional interpolation of a data tables

2. calculating the gradient matrix from the vertices of the above grid

where this burden increases disproportionately with the dimension or the number of independent variables of the table. In practice it was possible to eliminate the first and heavily accelerate the second, by transforming the independent variables of the tables into indexes with the aid of some simple reversible auxiliary equations. In this way, the location of the required point in the grid is directly determined, and since the grid points are now all a unit distance apart, all the divisions are eliminated in calculating the gradient matrix.

The resulting *Modified Newton-Raphson Routine* was implemented in *DynHist* and *DynCarp* programs, which for the five dimensional tables tried so far (i.e. data table with 5 independent variables) are almost as fast as normal operation with high speed engine models. Although the graphical display of *DynCarp* and *DynHist* are not quite as detailed or fancy as in data animation/visualization packages, their speed of animation is much faster than anything the author has seen at demonstrations or software used by the university. The majority of these packages are sluggish on a low to middle range workstations and cannot be considered real time.

Speed of data animation or function generation from data has in recent years attracted a lot of attention both in developing a new techniques/software and new hardware. For example, *Silicon Graphics* has been looking at efficient function generation techniques and programming them onto a special chip, in order to increase speed of function generation.

8.3.4 Calculation of Steady State Off-Design Performance of Engines

The calculation of off-design performance is actually a standard practice and can be found in most propulsion text books. It is an iterative process to determine a particular combination of thermodynamic variables, so that mass, momentum and energy equations are satisfied. In practice, these equations form n simultaneous equations in n unknowns (i.e. $n \times n$ system) which are solved very accurately, usually by the Multivariate Newton-Raphson (*N-R*) technique. The rigorous methods use a component map for each component (e.g. compressor, burner, turbine etc.) which are interrogated at every iteration step to find the unmatched operating point of each component which allow the calculation of the error matrix. After a number of steps, the sum of the squares of the error tends to zero at which point a steady state off-design point is found. The availability and accuracy of the components maps (especially the compressor or fan map) is generally a problem and there are three programs from NASA *LeRC*^[9] which

output more rigorous fan, compressor and turbine maps. The less rigorous version of the above would be to use simple mathematical models for each component map^[18].

Now an off-design point takes on average 10 iteration steps (from *GASTURB* for a two spool unmixed turbofan) and if the equilibrium running line (*ERL*) is to be approximated by 20 segments, then one complete calculation of the *ERL* will require 200 *N-R* steps. Although 200 *N-R* steps is considered by many analysts as *fast*, it will not be of order of milliseconds on a benchmark workstation. It is important to remember that one *N-R step* involves interrogation of the component maps as well as updating the $n \times n$ gradient matrix. *GASTURB 5.1* takes about 30 seconds to calculate the *ERL* of a two spool turbofan on a 486 DX-50 MHz personal computer.

The author recently became informed of a trick to speed up the above process (courtesy of NASA LeRC) by not updating the $n \times n$ gradient matrix as long as the sum of the squares of the error is decreasing. While this does not improve the number of iteration steps, it does save on calculating the gradient matrix. This trick works because as it turns out the *ERL* of most engines are not very curvy overall. This is especially so in the small segments of the *ERL*.

Now the number of iterations of the *N-R* scheme is related to the first good guess of the next point on the *ERL*. The better the first guess the smaller the number of iterations that is required for convergence. It follows that if one had a direct non-iterative technique for determining the next point on the *ERL* to a high accuracy (i.e. 96% to 99% accurate), then the *N-R* would not require more than one or two steps to converge. Fortunately the direct non-iterative off-design solutions given in section 4.4 do possess this level of accuracy and they can therefore be used to guide and accelerate the rigorous iterative off design method. Assuming this guidance reduces the number of steps to two per off design point, then the whole of the *ERL* (for 20 segments) would require only 40 iterative steps compared with 200 from above.

In summary, combining the following tricks

1. 96%-99% accurate first guess of the next point on *ERL* from the direct non-iterative off-design solutions derived in chapter 4
2. not updating the gradient matrix as long as the sum of the squares of the errors is decreasing
3. the modifications applied to the *N-R* (section 6.6.)

results in drastic improvements in the calculation effort of steady state off-design performance of engines.

It should be mentioned again that *Cockshutt*^[7] first derived the closed form solution for a simple turbojet spool and discussed their potential for guiding a rigorous solution, but his result was too simple to be of practical use (see section 4.7 on page 95).

8.3.5 Execution Speed of *CAGED* Engine Models

Speed of execution is a critical issue for the *NDC* concept and hence for the engine models generated by *CAGED*, as mentioned very often in this thesis. It is therefore desirable to measure the execution speed of the engine models and the overall design optimization cycle in the current implementation. However this is a difficult task as execution times are of order of milliseconds and external hardware (e.g. a signal generator) would be required.

So the only approximate measure that we have at the moment is the fact that one *NDC* loop appears to be instantaneous to the user. This implies that the time it takes to calculate and display the results graphically is better than the movie rate 1/24 seconds.

The following table provides an approximate comparison between *CAGED* and *GASTURB*.

TABLE 4

Comparison of execution speeds between CAGED and GASTURB engine models

	<i>GASTURB</i> (PC 486 DX2 50 MHz)	<i>CAGED</i> (SGI Iris Indigo R2000A/R3000 33 MHz)
calculation of 400 on-design points	~ 62.7 s	< 1/24 s
calculation of the <i>ERL</i>	~ 31.3 s	< 1/24 s
total time required to update an on-design carpet plot with 25 points including graphical display	~ 100 s	< 1/24 s

Note that the two programs run on separate computers. For pure number crunching, the *PC* is faster than the *Indigo*. However the graphical speed of the *Indigo* is far superior.

The *GASTURB* executions times were measured by a simple stop watch and are an average of several runs. They are user dependent so that it is possible to improve on the reported times by several seconds by becoming more used to the program. However the improvement will not be of orders of magnitude.

Apart from being suitable for the *NDC*, high speed models facilitate the use of computation intensive and robust numerical optimization techniques (e.g. the adaptive grid, genetic algorithms, simulated annealing etc.).

Another application for high speed engine models is in optimal real time control, where it is very desirable to be able to instantly estimate performance/behaviour of the engine at some future instant in time. This information/anticipation can help to optimize the necessary control actions at the current instant in time.

8.4 Future Work

The nature of the current research program can be classified as theoretical. Several new theories and techniques were developed which aid the conceptual/preliminary design of gas turbine based propulsion systems. Given the very limited resources, man power, time, in-house propulsion knowledge base and computing support, not *all* the topics discussed in this report could be validated. Therefore the sequence of future tasks must begin with the validation of what remains unvalidated.

First, the “*Well Designed and Well Matched*” closed form solution for the off-design performance of a generic spool should be validated. Although the *restricted* and *unrestricted* solutions have been derived, it is very desirable to use a method which does away entirely with the question of restriction at the outlet of the generic spool. This technique requires information about the type of the first turbine stage (i.e. the degree of reaction at the design point) which is always easily available from the velocity triangles of the first turbine stage at the design point.

The technique for quasi-steady transient behaviour of a generic spool of section 4.5 on page 91 should then be validated which derives its high speed by eliminating the compressor/turbine map reading usually involved in this type of analysis. Once validated, the theory can easily be extended to calculate the controlled (or forced) transient response of the generic spool as compared to the natural response currently covered by the theory.

The non-iterative closed form nature of the off-design solutions allow *CAGED* to generate very high speed off-design performance models whose accuracy can vary between 96% (at part load) to 99% near the design point. Since the high speed on-design models have a demonstrated accuracy of around 99.9%, then this level of accuracy should also be attempted in the high speed off-design engine models also which is only possible by an iterative solution. As discussed previously, the number of

iterations and the calculation effort to find an off-design point can be drastically reduced by allowing the above closed form solutions guide the iteration and by applying the speed up tricks to the *Newton Raphson* techniques.

One important issue that has not been discussed in this thesis is *Control Schedules*. Control schedules have a very significant impact in the preliminary design of engines. Scheduling the turbine entry temperature for example, strongly influences the critical conditions (e.g. maximum temperature, shaft speeds etc.) reached within the tentative mission/flight envelope of the engine. Since these critical conditions determine the size and mechanical design of components, it is clear that control schedules should be included in preliminary design study. Control schedules become even more important as engine concepts become more complex employing more variable geometry components. The differential analysis of the steady state off-design performance of engines presented in this report, treats the *natural* or *uncontrolled* behaviour of engines. However as *Urban*^[65] has shown, differential analysis provides a very convenient means for inclusion of control schedules in off-design analysis. The author is therefore confident that the closed form solutions presented herein can be extended to include controlled off-design behaviour, which is an item in the list of future tasks.

The author has not yet been able to categorize his *Non-Random Adaptive Grid* technique into some well known class of methods about which a lot of information is available and has recently requested the reviewers of his publication^[57] to look into the matter. The theoretical derivations of the coupling between *Probability*, *Accuracy*, and *Calculation Effort* show that for a finite calculation effort (i.e. not infinite number of steps) this technique gives orders of magnitude better return on investment than any robust method in which the total search volume reduces and/or the search points are generated randomly.

It will be very advantageous if a proper category is found because then the quality of the technique will be well known as compared to other techniques. If a category is not found, then the theoretical derivations and illustrated success on a particularly deceiving merit function provides a strong motivation for applying this technique to a set of standard merit functions and compare the return on investment with other robust optimization techniques.

The above tasks are estimated to require a maximum of six months of full time research. Once completed, then the *Integrated Thermo-Mechanical* system can be set up depending on the availability of the *Boeing WATE*^[46] code and its extension to calculate engine costs^[47]. These codes may not have been written for maximum execution speed and may have to be slightly modified to remove the bottle necks.

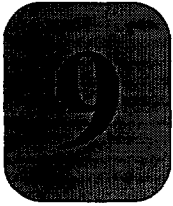
Once *CAGED* has been upgraded to include *Integrated Thermo-Mechanical*, then a very flexible and full conceptual/preliminary design package for propulsion systems is available which mimics the *NDC* and the really interesting tasks can begin, for example,

- design/optimize an engine and compare with existing engines in the same class
- input an existing engines and identify any possible improvements in weight, cost and/or performance

The engine models that are generated user-interactively with the *CAGED* system will be particularly useful to global design optimization of airframe/engine combinations because,

1. they have very high execution speeds allowing the use of global optimization methods for locating a true optimum
2. they contain the majority of information that may be required, namely engine thermodynamics, geometry, weight and costs

The author believes that this thesis together with the current operating version of *CAGED*, *DynCarp* and *DynHist* has identified the potential power of the envisaged design approach and provides sufficient information to justify and motivate the above list of future tasks.



References

Author	Title	Source	ID Number	date/ Place
1 Ashley, S.	"Engineous Explores the Design Space"	Mechanical Engineering	Vol. 114/No. 2	Feb. 1992.
2 Bailey M.W.	"Automated Aircraft Engine Costing Using Artificial Intelligence"	GE	AIAA 90-1887	26 th Joint Prop. Conf., July 16-18, 1990
3 Bill, C.	"Development and application of a computer-based system for conceptual aircraft design"	Ph.D. Thesis, Delft University Press	ISBN 90-6275-484-8	Delft, November 1988
4 Chapel M. S. & Cockshutt E. P	"Gas-turbine Cycle Calculations: thermodynamic data tables for air and combustion products"	National Research Council of Canada	NRC No. 14300	Ottawa, 1974

References

Author	Title	Source	ID Number	date/ Place
5 Chapel M. S. & Cockshutt E. P.	"Thermodynamic Data Tables for Air & Combustion Products"	National Research Council of Canada	Aeronautical Report LR-517	Ottawa
6 Churchhill R. V., Brown J. W.	" <u>Fourier Series and Boundary Value Problems</u> "	McGraw-Hill	ISBN 0-07-010881-1	1987
7 Cockshutt, E. P.	"Gas Turbine Cycle Calculations: Differential Methods in the Analysis of Equilibrium Operation"	Canadian Aeronautics and Space Transactions	Vol. 1, No. 1	March 1968
8 Cohen H., Rogers G.F.C., and Saravanamuttoo, H.I.H.	"Gas Turbine Theory"	Longman	ISBN-0-582-44926	1985
9 Converse, G. L. and Griffin R. G.	"Extended Parametric Representation of Compressors, Fans and Turbines" Volumes I, II, III	General Electric Co.	NASA CR-174645 NASA CR-174646 NASA CR-174647	1984.
10 Cully D. G., and Gunness, R. C.	"A Cost Modeling Approach to Engine Optimization"	Garret Turbine Engine Company	AIAA 82-1185	1982
11 Cyrus, J. D.	"Engine Component Life Prediction Methodology for Conceptual Design Investigations"	Naval Air Development Center	ASME 86-GT-24	June 8-12, 1986

Author	Title	Source	ID Number	date/ Place
12 Cyrus J.D, Onat E.	"A Pre-Design Code for predic- itng Engine Acquisition Costs"	Naval Air Dev. Center & Boeng Military Air- plane Dev.	AIAA 80- 0055	Jan. 14-16/ Pasadena, Cali- fornia
13 Fielding D., Topps J. E. C.	"Thermodynami c Data for the Calculation of Gas-Turbine Performance, R&M 3099"	Aeronautical Research Com- mittee	R&M 3099	UK, 1959
14 Fishbach, L. H.	"Computer Sim- ulation of Engine Sys- tems"	AIAA 18th Aer- ospace Sciences Meeting	AIAA 80- 0051	January 14-16, 1980, Pasadena, Cali- fornia.
15 Fishbach, L. H.	"PREPWATE- An interactive Preprocessing Computer Code to the Wate Analysis of Tur- bine Engines (WATE) Com- puter Code"	NASA Lewis, OH.	NASA TM- 83545	December, 1983
16 Fishbach, L. H. and Caddy, M.	"NNEP- The Navy NASA Engine Program "	NASA Lewis, OH.	NASA-TM X- 71857	1975.
17 Fishbach, L. H. and Gordon, S.	"NNEPEQ- Chemical Equi- librium Version of the Navy NASA Engine Program"	NASA Lewis, OH.	NASA TM- 100851	1988.
18 Flack, R. D.	"Analysis & Matching of Gas Turbine Components"	University of Virginia, Char- lottesville, VA	International Journal of Turbo and Jet Engines, vol 7	1990

References

Author	Title	Source	ID Number	date/ Place
19 Frank Watts ,A.	"Aircraft Turbine Engines Deveopment and Procurement Costs"	RAND	Memorandum RM-4670-PR	Nov. 1965
20 Ganji, A. R. et. al.	"Aerothermodynamic Analysis of Combined Cycle Propulsion Systems"	28th Aerospace Sciences Meeting	AIAA 90-0090	Reno, Nevada
21 Geiselhart, M. J. et. al.	"Computer Program for Estimating Performance of Air-Breathing Engines"	NASA Langly. VA.	NASA TM-4254	May 1991.
22 Gerend Robert P. and Roun-dhill	"Correlation of Gas Turbine Weight and Dimenaions"	The Beoing Company	AIAA-70-669	CA., 1970
23 Glassman, Arthur J	"Turbine Design and Application"	NASA	NASA SP-290, Vol. I, II & III	Washington, D.C. 1972
24 Goel S., Cherry, D., Gregory B	"knowledge based system for the preliminary aerodynamic design of aircraft engines"	GE		

Author	Title	Source	ID Number	date/ Place
25 Gordon, S. and, McBride, B.J	"Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman-Jouget Detonations"	NASA Lewis, OH.	NASA SP-273	1971
26 Gottfried, Byron S. and Weisman, Joel	"Introduction To Optimization Theory"	Prentice Hall, Inc.	ISBN 0-13- 491472-4	1973
27 Grigor'ev, S. B.	"Systems Approach to Engine Flow Path Configuration Design with the Aid of CADs"	Istesviya VUZAviationay a Tekhnica	Vol. 32, No. 1.	1989
28 Hill P. G. and Peterson C. R.	"Mechanics and Thermodynamics of Propulsion"	Addison-Wesley	ISBN-0-201- 02838-7	1970
29 Johnsen, I. A. and Bullock, R. O.	"Aerodynamic Design of Axial Flow Compressors"	NASA	NASA SP-36	1965
30 Kitson, Frederick L.	"Future Developments in Graphics Workstations"	CAD/CAM Symposium, "CONGRATULATIONS", Delft University of Technology"	Closing Lecture	28 January, 1992

References

Author	Title	Source	ID Number	date/ Place
31 Koenig, W. and Fishbach, L H.	"GENENG- A Program for Cal- culating Design and Off-Design Performance for Turbojet and Turbofan Engines"	NASA Lewis, OH.	NASA TN D- 6552	February 1972.
32 Kolb, Mark A. and Bailey , Michael W	"FRODO: Con- straint Based Object-Model- ling For Prelimi- nary Design"	GE		
33 Korakianitis, T. et. al.	"models for Pre- dicting the Per- formance of Brayton-Cycle Engines"	ASME-IGTI International Congress and Exposition	ASME 92- GT-361	June 1-4, 1992, Cologne, Ger- many;
34 Kott et. al.	"An Autono- mous Artificial Designer of Thermal Energy Systems"	<u>ASME Journal of Engineering for Gas Turbine and Power</u>		
35 Kurzke, J	"Calculation of Installation Effects within Performance Computer Pro- grams"	<u>"Steady and Transient Per- formance Pre- diction of Gas Turbine Engines"</u>	AGARD-LS- 183	May 1992.
36 Kurzke, J	"GASTURB- A Program for Cal- culating Design and Off-Design Performance of Gas Turbines"	Author	Version 5.2	July 1993, Munich, Ger- many.

Author	Title	Source	ID Number	date/ Place
37 Lamekin, M., Bogdanov, A.S.	Influence of Bypass ratio on Jet Engine Weight"	Izvestia VUZ. Aviatsionnaya Technika	vol. 21, No. 1, pp. 68-73	1973
38 Lee, H., Goel, S., Siu S. Tong, Gregory, B., Hunter S.	"Toward Model- ling the Concur- rent Design of Aircraft Engines"	General Electric	ASME 93- GT-193	May 24-27, 1993
39 Lefebvre, Arthur H	" <u>Gas Turbine Combustion</u> "	Hemisphere, New York		1983
40 Mattingly, Jack D., Heiser, Wil- liam H.	"Improvements in Teaching Air- craft Engine Design"	28th AIAA/ ASME/SAE Joint Propulsion Conference	AIAA 92- 3758	July 6-8, 1992, Nashville, TN.
41 Mattingly, J.	"Airbreathing Propulsion Edu- cation Software for PCs"	24th AIAA/ ASME/SAE Joint Propulsion Conference	AIAA 88- 2970	July 11-13, 1988, Boston, MA.
42 Middel, J	"Development of a computer assisted toolbox for aerodynamic design of aircraft at subcritical conditions with application to three-surface and canard air- craft"	Ph.D. Thesis, Delft University Press	ISBN 90- 6275-768-5 / CIP	April, 1992
43 Miller, B. A. et. al.	"A Perspective On Future Directions in Aerospace Pro- pulsion System Simulation"	Fourth Interna- tional Confer- ence on Supercomputing	NASA TM 102038	April 30, 1989., Santa Clara, California.

References

Author	Title	Source	ID Number	date/ Place
44 Munzburg H. G., Kurtzke J	<u>"Gasturbinen - Betriebsverhalten und Optimierung"</u>	Springer - Verlag, Berlin Heidelberg New York	ISBN 3-540-08032-5	1977
45 Murphy, Richard Lee	"An eclectic technological index as a determinant of R&D and production costs of jet aircraft engine"	University of Cincinnati	Ph.D. Thesis from UMI Order Number 9108581	1990
46 Onat, E. and Klees, W.	"A Method to Estimate Weight and Dimensions of Gas Turbine Engines"	Boeing Military Airplane Development	NASA CR 159481	1979.
47 Onat, E. and Tolle F. F.	"An extension of Engine Weight Estimation Techniques to Computer Engine Production Costs"	Boeing Military Airplane Development	NADC-78103-60	1979
48 Osmer J., Blevins, G.	"Life and Utilization Criteria in Design For Balanced Life and Performance"	P&W, and Wright Aero. Lab.	AIAA-80-1082	Connecticut, June 30, 1980

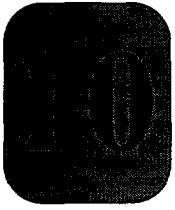
Author	Title	Source	ID Number	date/ Place
49 Palmer, J.R and Yan Cheng-Zhong,	"TURBOTRAN S- A Program- ming Language for the Perform- ance Simulation of Arbitrary Gas Turbine Engines with Arbitrary Con- trol Systems"	ASME-IGTI International Congress and Exposition	ASME 82- GT-200	1982
50 Pera, R. J., Onat E., Kless, G.W., Tjonne- land E.	"A Method to Estimate Weight and Dimansions of Gas Turbine Engines, Vol- ume 2: User's Manual"	NASA Lewis	NASA CR- 135171	1977
51 Pickerel	"Propulsion"	Rolls Royce, 45 th Oxford Air Transport Course	PNR90472	March 1988
52 Press, W.H., Flannery, B. P., Teukilsky, S. A. and Vetterling, W.T.	" <i>Numerical Rec- ipes in Pascal</i> , The Art of Sci- entific Comput- ing"	Cambridge Uni- versity Press	ISBN 0-521- 37516-9	1989
53 Roy-Aikins, J. E. A. et al.	" VATEMP - The variable Area Turbine Engine Match- ing Program"	ASME-IGTI International Congress and Exposition	ASME 90- GT-192	June 11-14, 1990, Brussels, Belgium.
54 Ruffles P. C.	"Reducing the Cost of Aero Engine Research & Development"	Rolls Royce	PNR 90341	August 1986

Author	Title	Source	ID Number	date/ Place
55 Rumyanstev, S..	"Engine and Airplane Matching Technique in the APPA-RAT High Level Automated Design System"	<u>Istesviya VUZAviationay a Tekhnica.</u>	Vol. 24, No. 3	1989
56 Sellers, James F. and Daniele, Carl J.	"DYNGEN- A Program for Calculating Steady State and Transient Performance of Turbojet Engines"	NASA LeRC	NASA TN D-7901	April, 1975.
57 Shahroudi, K.E.	"A Modern Computer Aided Approach to Conceptual/Preliminary Design of Gas Turbines"	ASME-IGTI International Congress and Exposition	ASME 94-GT-458	June 13-16, 1994.
58 Shahroudi, K.E.	<i>"Non-Random Adaptive Grid Method for High Speed Optimization of Highly Dimensional, badly behaving Real Time Functions"</i>	ASME-IGTI International Congress and Exposition	ASME 94-GT-487	June 13-16, 1994.
59 Sugiyama, N.	"Generalized High Speed Simulation of Gas Turbine Engines"	ASME-IGTI International Congress and Exposition	ASME 90-GT-270	June 11-14, 1990, Brussels, Belgium.

Author	Title	Source	ID Number	date/ Place
60 Szuch, J. R.	"HYDES- A Generalized Hybrid Computer for Studying Turbojet and Turbofan Engine Dynamics"	NASA	NASA TM X3014	1974
61 Tong, Siu s, Powel D. and Goel S	"Integration of Artificial Intelligence and Numerical Optimization Techniques for the Design of Complex Aerospace Systems"	1992 Aerospace Design Conference	AIAA 92-1189	Feb. 3-6, 1992, Irvine, CA.
62 Torella, G.	"Simulation of Performance, Behavior, and Faults of Aircraft Engines"	23rd AIAA/ ASME/SAE Joint Propulsion Conference	AIAA-87-1868	June 29 - July 2, 1987, San Diego, CA.
63 Torenbeek, E.	"Analytical Method for Computing Turbo-Engine Performance At Design and Off-Design Conditions"	Aerospace Engineering Dept., Delft TU	Memorandum M-188	January 1973.
64 Torenbeek, E.	"Synthesis of Subsonic Airplane Design"	Delft University Press	ISBN 90-247-2724-3	1982
65 Urban, Louis A.	"Gas Turbine Engine Parameter Interrelationship"	Hamilton Standards	Secound Edition	1969
66 Von Ohain, H.	"The Next 50 years, Aircraft Engines I"	Exxon Air World	Vol 40, No2	1988

References

Author	Title	Source	ID Number	date/ Place
67 Wilson, D.G.	"The Design of High Efficiency Turbomachinery and Gas Turbines"	MIT Press	ISBN-0-262-23114-X	1984
68 Wittenberg, H.	"Prediction of Off-Design Performance of Turbojet and Turbofan Engines"	AGARD	CP-242 Paper 4	1978
69 Wittenberg, H	"Prediction of Off-Design Performance of Turbo-Shaft Engines"	Vertica	Volume 8-No. 3., pp. 197-208	1984



Appendix

TABLE 5

(Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Mattingly and Heiser ^[40]	High Bypass Ratio Turbofan	$\theta_0 \leq TR \rightarrow \alpha = \delta_0 (1 - 0.49 \sqrt{M_0})$ $\theta_0 > TR \rightarrow \alpha = \delta_0 \left(1 - 0.49 \sqrt{M_0} - \frac{3(\theta_0 - TR)}{1.5 + M_0} \right)$	$(0.5 + 0.6M_0) \sqrt{\theta}$
	Low Bypass Ratio Turbofan	<p><u>Maximum Power</u></p> $\theta_0 \leq TR \rightarrow \alpha = \delta_0$ $\theta_0 > TR \rightarrow \alpha = \delta_0 \left(1 - \frac{3.5(\theta_0 - TR)}{\theta_0^2} \right)$ <p><u>Military Power</u></p> $\theta_0 \leq TR \rightarrow \alpha = 0.6\delta_0$ $\theta_0 > TR \rightarrow \alpha = \delta_0 \left(1 - \frac{3.8(\theta_0 - TR)}{\theta_0^{1.6}} \right)$	<p><u>Maximum Power</u></p> $(1.8 + 0.3M_0) \sqrt{\theta}$ <p><u>Military and LowPower</u></p> $(1.0 + 0.35M_0) \sqrt{\theta}$

TABLE 5

(Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Mattingly and Heiser ^[40]	Turbojet	<p><u>Maximum Power</u></p> $\theta_0 \leq TR \rightarrow \alpha = \delta_0 (1 - 0.3 (\theta_0 - 1) - 0.1 \sqrt{M_0})$ $\theta_0 > TR \rightarrow \alpha = \delta_0 \left(1 - 0.3 (\theta_0 - 1) - 0.1 \sqrt{M_0} - \frac{1.5 (\theta_0 - TR)}{\theta_0^2} \right)$ <p><u>Military Power</u></p> $\theta_0 \leq TR \rightarrow \alpha = 0.8 \delta_0 (1 - 0.16 \sqrt{M_0})$ $\theta_0 > TR \rightarrow \alpha = 0.8 \delta_0 \left(1 - 0.16 \sqrt{M_0} - \frac{24 (\theta_0 - TR)}{(9 + M_0) \theta_0^2} \right)$	<p><u>Maximum Power</u></p> $(1.3 + 0.35 M_0) \sqrt{\theta}$ <p><u>Military and Low Power</u></p> $(1.7 + 0.26 M_0) \sqrt{\theta}$
	Turbo-prop	$M_0 \leq 0.1 \rightarrow \alpha = \delta_0$ $\theta_0 \leq TR \rightarrow \alpha = \delta_0 (1 - 0.96 (M_0 - 0.1)^{0.25})$ $\theta_0 > TR \rightarrow \alpha = \delta_0 \left(1 - 0.96 (M_0 - 0.1)^{0.25} - \frac{3 (\theta_0 - TR)}{8.13 (M_0 - 0.1)} \right)$	$(0.2 + 0.9 M_0) \sqrt{\theta}$

TABLE 5

(Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Torenbeek ^[63]	Turbofan	$\delta \frac{\Psi}{\Psi_{TO}} \mu^{1.5} \left(\frac{1}{\sqrt{\Phi_{TO}}} + 0.6 \frac{(\Phi_{TO} - 1)}{\Phi_{TO}} \mu^{1.5} \left(\frac{\Psi}{\Psi_{TO}} + \frac{34.714M}{\Psi_{TO}} \right) \right)$ <p>Where the following are calculated at design point,</p> $\Psi = \frac{T}{W\sqrt{\theta}} = 34.714 (\sqrt{5\eta_n G} - M), \Phi_{TO} = 1 + \frac{T_{TO}}{A_e P_\infty}$ $\mu = 1 + 0.2M^2, \phi = T_{t_4}/T_0, \kappa = \mu (\epsilon_c^{0.2857} - 1)$ $G = \left(\phi - \frac{\kappa}{\eta_c} \right) \left(1 - \frac{1.011}{\eta_i^{0.2857} (\kappa + \mu) \left(1 - \frac{\kappa/\phi}{\eta_c \eta_t} \right)} \right)$	$(0.711\sqrt{\theta}) \left(\phi - \mu - \kappa/\eta_c \right) \sqrt{5\eta_n (1 + \eta_{tf}\lambda) \left(G + 0.2M^2 \frac{\eta_i \lambda}{\eta_{tf}} \right) - (1 + \lambda)M}$
	Turbo-prop	<p>Specific Equivalent Shaft Horse Power (hp/N/s)</p> $\frac{P_{eq}}{W\theta} = 382 \left(\eta_i G - M^2 \left(0.4 - \frac{0.193}{\eta_i} \right) \right)$	$C_p = \frac{\text{fuel consumption/hr}}{\text{brake horse power}}$ $0.086 \frac{\left(\phi - \mu - \frac{\kappa}{\eta_c} \right)}{\left(\eta_i G - 0.28 \frac{M^2}{\eta_i} \right)}$ <p>(N/hr/kw)</p>

TABLE 5

(Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

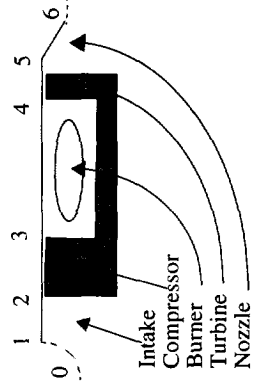
Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Wittenberg [68] [69]	Turbojet	<p>Choked Exhaust Nozzle</p> $\epsilon_C = \left[1 + \phi_1 \left(\frac{\gamma_a - 1}{\gamma_a \eta_{c_{design}}} \epsilon_{c_{design}} - 1 \right) \right]^{\frac{\gamma_a \eta_c}{\gamma_a - 1}}$ $\frac{m_a \sqrt{T_2}}{P_{t_2}} = \left(\frac{m_a \sqrt{T_2}}{P_{t_2}} \right)_{design} \frac{1}{\sqrt{\phi_1}} \frac{\epsilon_c}{\epsilon_{c_{design}}}$ $\frac{\phi}{\phi_{design}} = \frac{\epsilon_c}{\epsilon_{c_{design}}}$ <p>Where</p> $\epsilon_C = \text{Compressor Pressure Ratio}, \phi = \frac{\frac{T_g}{P_0 A_6} + 1}{P_{t_2}/P_0}$ $\phi = \frac{T_{t_4}/T_{t_2}}{\left(T_{t_4}/T_{t_2} \right)_{Design}}$	$\frac{m_F}{P_{t_2} \sqrt{T_{t_2}}} = \frac{C_{p_m} m_a \sqrt{T_2}}{\eta_b H P_{t_2}}$ $\times \left(\frac{T_{t_4}}{T_{t_2}} - \epsilon_c \frac{\gamma_a - 1}{\gamma_a \eta_c} \right)$ 

TABLE 5 (Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

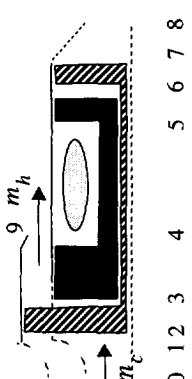
Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Wittenberg [68] [69]	Turbofan	$\epsilon_f = \left(\frac{T_{t_3}}{T_{t_2}} \right)^{\frac{\eta_f \gamma_a}{\gamma_a - 1}}$ $\epsilon_{HPC} = \left[1 + \Phi_2 \left(\frac{\gamma_a - 1}{\gamma_a \eta_{C_{design}} - 1} \right) \right]^{\frac{\gamma_a \eta_{C_{design}}}{\gamma_a - 1}}$ <p>where $\lambda = \frac{m_c \sqrt{T_{t_3}}}{P_{t_3}} / \frac{m_h \sqrt{T_{t_3}}}{P_{t_3}}$</p> $\frac{1}{\Phi} \frac{\lambda + 1}{\lambda_{des} + 1} \left(\frac{1 - T_{t_2}/T_{t_3}}{1 - (T_{t_2}/T_{t_3})_{des}} \right) = \frac{1 - T_{t_7}/T_{t_6}}{1 - (T_{t_7}/T_{t_6})_{des}}$	

TABLE 5

(Semi-)Analytical Off-Design Relations For Various Types of Engines. Symbols from original references used in table, hence not applicable elsewhere in this report.

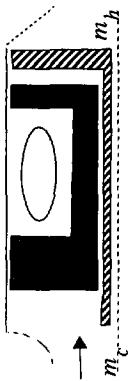
Author	Engine Type	Thrust Lapse $\alpha = T/T_{SLS}$	Thrust Based SFC
Wittenberg [68] [69]	Turbo-prop or Turbo-shaft	$\frac{\frac{P_{br}}{P_{t_2} \sqrt{T_{t_2}}}}{\left(\frac{P_{br}}{P_{t_2} \sqrt{T_{t_2}}} \right)_{des}} = \frac{\epsilon_c}{\epsilon_{c_{des}}} \sqrt{\phi} \frac{1 - \left(\frac{P_{t_6}}{P_{t_5}} \right)^{\eta_t \left(\frac{\gamma_g - 1}{\gamma_g} \right)}}{\left(1 - \left(\frac{P_{t_6}}{P_{t_5}} \right)^{\eta_t \left(\frac{\gamma_g - 1}{\gamma_g} \right)} \right)_{des}}$ <p>where for perfect expansion:</p> $\frac{P_{t_5}/P_{t_6}}{\left(P_{t_5}/P_{t_6} \right)_{des}} = \frac{\epsilon_c}{\epsilon_{c_{des}}} \frac{P_{t_2}/P_{t_0}}{\left(P_{t_2}/P_{t_0} \right)_{des}}$ <p>and</p> $P_{t_2}/P_{t_0} = \eta_R \left(1 + \frac{\gamma_a - 1}{\gamma_a} M_0^2 \right)$	

TABLE 6

**Summary of Closed Form Direct Analytical Off-Design Relations
for the Generic Spool**

		Type Of Generic Spool		
		Restricted Outlet	Outlet Matched to Expanding Nozzle	Well Designed Well Matched Spool
Variable				
Z		1	1	2
c		0	$C = \frac{k-1}{(2a_c k - 1) (\phi - \tau_s) / \tau_s}$ $k = 1 - \frac{1}{\nabla_{\mu_{in}} \pi_n}$	0
B		$1 - \tau_c / (2a_c (\tau_c - 1))$		1
Pressure Recovery π_r	X_{π_r}	0	0	0
	Y_{π_r}	0	$\nabla \tau_r \left(\frac{1}{2} - a_c k \right)$	0
m_{rat} from bypass nozzle	$X_{m_{rat}}$	$\frac{1}{2} \frac{1 - m_{rat}}{m_{rat}} \left(\frac{(2a_n - 2) + \tau_n (1 - 2a_n)}{2a_n (1 - \tau_n)} (X_{\pi_r} + 1) + \frac{1}{2a_c} - 1 \right)$		Previous column times $2(I-G)$
	$Y_{m_{rat}}$	$\frac{1}{2} \frac{1 - m_{rat}}{m_{rat}} \left(\frac{(2a_n - 2) + \tau_n (1 - 2a_n)}{2a_n (1 - \tau_n)} Y_{\pi_r} + 1 \right)$		Previous column times $2(I-G)$
General m_{rat} from elect. analogy	$X_{m_{rat}}$	$-\frac{1}{2} (1 - m_{rat}^2) \left(\frac{p_6/p_2}{1 - p_6/p_2} - \frac{p_4/p_2}{1 - p_4/p_2} \right)$		Previous column times $2(I-G)$
	$Y_{m_{rat}}$	0		0

TABLE 6

**Summary of Closed Form Direct Analytical Off-Design Relations
for the Generic Spool**

		Type Of Generic Spool		
		Restricted Outlet	Outlet Matched to Expanding Nozzle	Well Designed Well Matched Spool
π_b	X_{π_b}	$-2 \left(\frac{\pi_b - 1}{\pi_b} \right) \times \left(\frac{1}{2a_c} - 1 + X_{m_{rat}} \right)$	$-2 \left(\frac{\pi_b - 1}{\pi_b} \right) \left(\frac{1}{2a_c} - 1 + X_{m_{rat}} \right) \times (1 + C)$	Same as Restricted Outlet
	Y_{π_b}	$-2 \left(\frac{\pi_b - 1}{\pi_b} \right) (1 + Y_{m_{rat}})$	$-2 \left(\frac{\pi_b - 1}{\pi_b} \right) (1 + Y_{m_{rat}}) (1 + C)$	Same as Restricted Outlet
Internal Spool Pressure Ratio π_b	X_{π_b}	$-\nabla_{\mu_2} \pi_b \times \left(\frac{1}{2a_c} - 1 + X_{m_{rat}} \right)$	$-\nabla_{\mu_2} \pi_b \left(\frac{1}{2a_c} - 1 + X_{m_{rat}} \right) \times (1 + C)$	Same as Restricted Outlet
	Y_{π_b}	$-\nabla_{\mu_2} \pi_b \times (1 + Y_{m_{rat}})$	$-\nabla_{\mu_2} \pi_b (1 + Y_{m_{rat}}) (1 + C)$	Same as Restricted Outlet
Correction For $\nabla a_c, P_{OffTake}$		$(\nabla \pi_c)_{intermediate} = \nabla \pi_c - \nabla a_c \log(\pi_c)$ $(\nabla \pi_c)_{corrected} = (\nabla \pi_c)_{intermediate} \left(\frac{\tau_c}{\tau_c + \frac{P_{OffTake}}{m_1 C_p T_1}} \right)^{a_c}$		N/A
$\nabla \emptyset$		$2(-1 + (\nabla \pi_c)_{corrected} + \nabla \pi_b - \nabla m_{rat})$		$2(1 - G \nabla m_{rat})$
$\nabla \pi_s$		$\nabla \pi_c \left(\frac{2a_t(\emptyset - \tau_s)}{\tau_s} \left(1 - \frac{\tau_c}{\tau_c - 1} \frac{1}{2a_c} \right) + 1 \right) + \nabla \pi_b \left(\frac{2a_t(\emptyset - \tau_s)}{\tau_s} + 1 \right) + \left(\frac{2a_t(\emptyset - \tau_s)}{\tau_s} \right) \left(-1 - \frac{\nabla m_{rat}}{2} \right)$		

TABLE 7

Test of convergence for the original Non-Random Adaptive Grid

s=5, optimum is at 0.7750000, 0.1500000, 0.5250000									
ϵ	P	$N_{s_{min}}$	$N_{s_{max}}$	F	x	y	z	Converged?	ϵ achieved
5e-06	1.000000e+00	724	2095	1099.53	0.775	0.149882	0.525	y	3.920055e-05
1e-05	1.000000e+00	456	1368	1099.26	0.775	0.149816	0.525	y	6.134613e-05
1.5e-05	1.000000e+00	348	1066	1099.26	0.775	0.149815	0.525	y	6.170000e-05
2e-05	1.000000e+00	287	893	1099.83	0.775	0.149956	0.525	y	1.451378e-05
2.5e-05	9.999995e-01	247	779	1099.35	0.775	0.149837	0.525	y	5.427949e-05
3e-05	9.999972e-01	219	696	1096.41	0.775	0.149102	0.525	y	2.991746e-04
3.5e-05	9.999888e-01	198	633	1097.28	0.775	0.149319	0.525	y	2.270630e-04
4e-05	9.999671e-01	181	583	1099.52	0.775	0.14988	0.525	y	4.005763e-05
4.5e-05	9.999212e-01	167	543	1098.18	0.775	0.149546	0.525	y	1.514270e-04
5e-05	9.998376e-01	156	509	1097.7	0.775	0.149426	0.525	y	1.913787e-04
5.5e-05	9.997010e-01	146	480	1097.23	0.775	0.149307	0.525	y	2.310651e-04
6e-05	9.994957e-01	138	455	1096.39	0.775	0.149101	0.525001	y	2.991926e-04
6.5e-05	9.992063e-01	131	433	747.441	0.7657	0.149369	0.524095	y	3.622718e-03
7e-05	9.988184e-01	124	413	1097.07	0.775	0.149268	0.525	y	2.439874e-04
7.5e-05	9.983195e-01	119	396	1038.14	0.775	0.149632	0.523282	y	6.952760e-04
8e-05	9.976990e-01	114	381	997.765	0.775	0.149553	0.522142	y	1.101763e-03
8.5e-05	9.969480e-01	109	367	651.898	0.7623	0.14984	0.524948	y	4.300279e-03
9e-05	9.960599e-01	105	354	1091.19	0.775	0.147798	0.525	y	7.340482e-04
9.5e-05	9.950301e-01	101	343	449.19	0.7565	0.149948	0.525004	y	6.178111e-03
0.0001	9.938554e-01	98	332	1094.78	0.775	0.148698	0.525002	y	4.332781e-04
0.000105	9.925345e-01	95	322	579.185	0.7675	0.149685	0.515423	y	5.796492e-03
0.00011	9.910673e-01	92	313	502.519	0.7580	0.149838	0.525017	y	5.703025e-03
0.000115	9.894550e-01	89	305	1096.19	0.775	0.14906	0.525006	y	3.113226e-04
0.00012	9.876998e-01	87	297	1094.07	0.775	0.148518	0.525	y	4.940143e-04
0.000125	9.858046e-01	84	289	974.184	0.775	0.149995	0.521426	y	1.192965e-03
0.00013	9.837732e-01	82	283	905.822	0.775	0.149777	0.519504	y	1.906289e-03
0.000135	9.816097e-01	80	276	899.692	0.775	0.149647	0.519342	y	2.003706e-03
0.00014	9.793188e-01	78	270	290.15	0.7533	0.148333	0.520894	y	9.158241e-03
0.000145	9.769055e-01	76	264	564.763	0.7603	0.14913	0.524179	y	5.470347e-03
0.00015	9.743750e-01	75	259	1030.38	0.7731	0.149785	0.525	y	7.231725e-04
0.000155	9.717327e-01	73	254	1093.21	0.775	0.148507	0.525094	y	4.666534e-04
0.00016	9.689838e-01	71	249	1098.17	0.775	0.149655	0.525051	y	9.796583e-05

TABLE 7

Test of convergence for the original Non-Random Adaptive Grid

s=5, optimum is at 0.7750000, 0.1500000, 0.5250000									
ϵ	P	$N_{s_{min}}$	$N_{s_{max}}$	F	x	y	z	Converged?	ϵ achieved
0.000165	9.661340e-01	70	244	518.645	0.7639	0.147477	0.516764	y	7.272063e-03
0.00017	9.631887e-01	69	240	1092.87	0.775	0.148283	0.52503	y	5.622846e-04
0.000175	9.601531e-01	67	235	1097.98	0.775	0.149609	0.525052	y	1.130831e-04
0.00018	9.570326e-01	66	231	457.546	0.7655	0.149941	0.512412	y	7.375482e-03
0.000185	9.538325e-01	65	227	1098.41	0.775	0.149656	0.525024	y	1.064046e-04
0.00019	9.505577e-01	64	224	327.585	0.25	0.148026	0.525018	n	-
0.000195	9.472131e-01	63	220	826.213	0.775	0.149389	0.517274	y	2.778791e-03
0.0002	9.438036e-01	62	217	1099.89	0.7750	0.149998	0.525009	y	2.206633e-06
0.000205	9.403336e-01	61	213	579.84	0.7733	0.14937	0.511195	y	5.369539e-03
0.00021	9.368076e-01	60	210	1098.53	0.775	0.149734	0.525046	y	7.339941e-05
0.000215	9.332299e-01	59	207	468.881	0.7732	0.14969	0.50787	y	6.391780e-03
0.00022	9.296045e-01	58	204	1094.99	0.775	0.148852	0.525047	y	3.668797e-04
0.000225	9.259354e-01	57	202	464.855	0.775	0.148481	0.507029	y	6.496577e-03
0.00023	9.222262e-01	56	199	398.031	0.7551	0.149371	0.525036	y	6.835407e-03
0.000235	9.184806e-01	55	196	521.328	0.775	0.149791	0.508572	y	5.545846e-03
0.00024	9.147018e-01	54	194	626.394	0.7617	0.149055	0.525056	y	4.758244e-03
0.000245	9.108933e-01	54	191	656.26	0.775	0.148076	0.512525	y	4.799616e-03
0.00025	9.070579e-01	53	189	928.2	0.7703	0.148495	0.525108	y	2.036524e-03
0.000255	9.031988e-01	52	187	287.776	0.2542	0.147953	0.521361	n	-
0.00026	8.993185e-01	52	184	458.216	0.775	0.149983	0.395732	n	-
0.000265	8.954199e-01	51	182	691.133	0.7635	0.148539	0.525023	y	4.315245e-03
0.00027	8.915053e-01	50	180	562.33	0.775	0.146357	0.50994	y	6.234432e-03
0.000275	8.875772e-01	50	178	1094.96	0.775	0.148757	0.525008	y	4.114512e-04
0.00028	8.836378e-01	49	176	579.86	0.7605	0.14726	0.524722	y	5.826482e-03
0.000285	8.796892e-01	49	174	451.387	0.775	0.14799	0.391715	n	-
0.00029	8.757335e-01	48	172	1091.39	0.775	0.148514	0.525305	y	3.936928e-04
0.000295	8.717725e-01	47	171	553.35	0.7612	0.149796	0.521963	y	5.688671e-03
0.0003	8.678080e-01	47	169	456.773	0.775	0.14988	0.394286	n	-
0.000305	8.638417e-01	46	167	1090.16	0.775	0.148169	0.525288	y	5.143760e-04
0.00031	8.598752e-01	46	165	690.965	0.7651	0.149822	0.522501	y	4.192545e-03
0.000315	8.559101e-01	45	164	136.068	0.2500	0.149005	0.506682	n	-
0.00032	8.519476e-01	45	162	1097.88	0.775	0.149571	0.525046	y	1.277492e-04
0.000325	8.479893e-01	44	161	984.41	0.7720	0.147581	0.525003	y	1.816948e-03

TABLE 8

Convergence test for the fast scheme, $N=5$, $\varepsilon = 0.01$, $s=5$, optimum at 0.775, 0.15, 0.525

P	$N_{s_{min}}$	$N_{s_{max}}$	steps to converg e	F	x	y	z	converge to 0.01?
5.532791e-01	20	20	20	1039.88	0.775	0.147046	0.523614	y
8.249307e-01	40	35	35	560.154	0.7641	0.148095	0.518299	n
9.018503e-01	60	44	44	311.26	0.2500	0.148205	0.523421	n
9.376576e-01	80	51	51	358.641	0.7597	0.14726	0.513913	n
8.952991e-01	100	58	43	1085.38	0.7748	0.148601	0.525328	y
9.079973e-01	120	63	45	855.201	0.768436	0.148826	0.526452	y
9.604833e-01	140	69	58	1088.03	0.774992	0.147135	0.525029	y
9.519461e-01	160	73	55	1085.41	0.774742	0.14893	0.525145	y
9.415823e-01	180	78	52	985.369	0.772415	0.146204	0.524695	y
9.519461e-01	200	82	55	937.233	0.777432	0.141293	0.523567	y
9.376576e-01	220	86	51	1004.13	0.774066	0.144161	0.529828	y
9.819701e-01	240	90	70	1089.9	0.774886	0.148953	0.525218	y
9.732970e-01	260	94	64	1023.24	0.773246	0.148402	0.524728	y
9.519461e-01	280	98	55	819.615	0.769038	0.147234	0.533733	y
9.749877e-01	300	101	65	892.966	0.770534	0.148845	0.531106	y
9.604833e-01	320	104	58	874.707	0.769767	0.143203	0.527582	y
9.841848e-01	340	108	72	961.045	0.772625	0.14693	0.530473	y
9.878337e-01	360	111	76	1036.44	0.776538	0.147854	0.524962	y

TABLE 9

Convergence test for the fast scheme, $N=5$, $\varepsilon = 0.001$, $s=2$, optimum at 0.76, 0.12, 0.51

P	$N_{s_{beam}}$	$\sqrt{s_{mz}}$	steps to converge	F	x	y	z	converge to 0.001?
9.495488e-01	81	60	60	131.438	0.250009	0.119973	0.373	n
9.841581e-01	131	78	78	136.759	0.250003	0.118583	0.39979	n
9.940196e-01	181	93	93	37.6394	0.25	0.75	0.397769	n
9.794818e-01	231	106	74	1074.59	0.759797	0.119759	0.510238	y
9.987530e-01	281	117	117	137.213	0.25	0.119704	0.396147	n
9.993936e-01	331	128	128	329.312	0.25	0.119771	0.51	n
9.996640e-01	381	137	137	526.103	0.753485	0.119883	0.510001	n
9.911639e-01	431	146	87	1040.6	0.759417	0.119576	0.510194	y
9.998970e-01	481	155	155	262.34	0.754622	0.119975	0.398322	n
9.999392e-01	531	163	163	281.009	0.25	0.119587	0.508184	n
9.986686e-01	581	170	116	1046.94	0.7596	0.119635	0.510676	y
9.999773e-01	631	178	178	329.404	0.25	0.119801	0.51	n
9.999857e-01	681	185	185	707.962	0.756701	0.119731	0.508456	n
9.999904e-01	731	191	191	459.326	0.76	0.11988	0.397533	n
9.997054e-01	781	198	139	1068.57	0.759828	0.119094	0.509914	y
9.998013e-01	831	204	145	1082.15	0.759871	0.119474	0.509985	y
9.999972e-01	881	210	210	939.557	0.759251	0.119962	0.508861	n
9.999981e-01	931	216	216	450.554	0.75262	0.119991	0.51	n
9.999988e-01	981	222	222	329.579	0.25	0.11986	0.51	n
9.999467e-01	1031	228	165	1090.27	0.759923	0.119842	0.509984	y
9.999994e-01	1081	233	233	474.393	0.752895	0.119923	0.51	n
9.999562e-01	1131	238	168	1080.02	0.759828	0.119863	0.50996	y
9.999997e-01	1181	244	244	537.703	0.753612	0.119965	0.51	n
9.999705e-01	1231	249	174	1072.25	0.759794	0.119875	0.509903	y
9.999998e-01	1281	254	254	950.213	0.76	0.119986	0.508299	n
9.999999e-01	1331	259	259	787.459	0.756449	0.11999	0.51	n
9.999875e-01	1381	264	187	1082.96	0.760141	0.119905	0.509957	y
9.999788e-01	1431	268	179	1015.79	0.759323	0.119536	0.509756	y
9.999947e-01	1481	273	200	1092.46	0.760056	0.119958	0.509975	y

ONTWIKKELING EN VALIDATIE VAN EEN COMPUTER-ONDERSTEUNDE ONTWERPMETHODIEK VOOR Vliegtuigmotoren gebaseerd op een GASTURBINE

Samenvatting

Het doel van dit promotie-onderzoek was het ontwikkelen van een modern en efficiënt gereedschap om ontwerpers te assisteren bij het mechanische en thermodynamische gedeelte van het conceptuele ontwerp van vliegtuigmotoren, de motor is gebaseerd op een gasturbine-cyclus. Vele van de arbeidsintensieve taken zijn geautomatiseerd; de intuïtieve taken en beslissingen worden echter overgelaten aan de menselijke ontwerper. Het ontwikkelde programma is aangeduid als Computer Aided General Engine Design, CAGED.

CAGED is erop gericht de "natuurlijke ontwerpfilosofie" na te bootsen. Dit wordt gedaan door middel van een snelle ontwerp-optimalisatiecyclus met een actieve beïnvloeding door de menselijke ontwerper als onderdeel van de cyclus. Hierdoor wordt het mogelijk voor de ontwerper om binnen korte tijd nieuwe ervaringen op te doen en conclusies te trekken. Dit komt in de plaats van de ondersteuning van het ontwerpproces met vuistregels. Hierbij wordt geen gebruik gemaakt van kunstmatige intelligentie.

Er zijn nieuwe methoden en concepten ontwikkeld om CAGED in staat te stellen met hoge snelheid motormodellen te genereren. De snelheid uit zich in een rekentijd van enkele milliseconden op een 33 MHz SGI Iris Indigo. Onder de nieuwe concepten en methoden bevinden zich de volgende:

1. Het concept van de universele component
2. Directe (niet iteratieve) en snelle analytische oplossing van de vergelijkingen buiten het ontwerppunt van een zogenaamde "generic spool"
3. Directe (niet iteratieve) en snelle methode voor het dynamische gedrag van een zogenaamde "generic spool"
4. Verwijdering van "Sequential Network Logic" van het gegenereerde motormodel
5. Alleen variabelen die van belang zijn voor het onder handen zijnde ontwerp worden opgenomen in het model
6. Gebruik van een snelle functiegeneratie voor de bepaling van stromingseigenschappen
7. Genereren van een motormodel in de vorm van een eenvoudige broncode

Deze motormodellen worden gegenereerd voor vrijwel iedere denkbare motorconfiguratie. De variabelen worden visueel gedefinieerd via de "Graphical User Interface". De korte rekentijd en de hoge nauwkeurigheid maken de modellen zeer geschikt voor opname in een groot optimalisatieprogramma, bijvoorbeeld in een vliegtuig-optimalisatieprogramma.

Er zijn twee nieuwe en efficiënte model-exploratie- en optimalisatiehulpgereedschappen werden ontwikkeld, *DynCarp* (Dynamic Carpets) en *DynHist* (Dynamic Histograms) genaamd. Deze

hulpmiddelen maken het mogelijk real-time door de multidimensionale ontwerpruimte te bewegen, terwijl er gebruik gemaakt wordt van snelle rekenmodellen. De ontwerper kan zo honderden variaties binnen één minuut doorrekenen. Tevens kan hij de betekenis en de consequenties van deze variaties bekijken in eenzelfde tijdsbestek. Dit is van groot nut bij het bestuderen van een nieuw concept of bij het aanbrengen van een grote wijziging in een bestaand concept waarbij er vooraf geen kennis is over het gedrag van het concept. Ook werden functiegeneratie-routines geschreven die met hoge snelheid een model van een gegevensbestand met een groot aantal onafhankelijke variabelen maken. Dit maakt het mogelijk om de routines DynCarp en DynHist tevens te gebruiken met gegevensbestanden die gegenereerd zijn door programma's die door derden zijn ontwikkeld.

Voor die gevallen waarin automatische numerieke optimalisatie gewenst is - met de menselijke ontwerper buiten de optimalisatiecyclus - werd een nieuwe optimalisatieroutine gebaseerd op het "Adaptive Grid" ontwikkeld waarbij dus niet gebruik hoeft te worden gemaakt van de functie-afgeleiden. Deze routine is geschikt voor de optimalisatie van functies met een groot aantal variabelen tot maximaal 25. De routine werkt met hoge snelheid; de rekentijd is enkele milliseconden. Verder werd analytisch de kwantitatieve koppeling afgeleid tussen de vereiste nauwkeurigheid, de kans dat het optimum werd gevonden en het aantal berekeningen. Deze is nuttig bij het bepalen van de tijd die nodig is om, voordat de oplossing wordt berekend, een resultaat met een gewenste nauwkeurigheid en kansdichtheid te verkrijgen.

De "Multivariate Newton-Raphson" rekenmethode wordt veelvuldig gebruikt in motor-analysesystemen vanwege de veelzijdigheid en eenvoud. De toepassing van eenvoudige numerieke trucs op deze methode resulteerde in een drastische verbetering in de rekensnelheid van het algoritme. Dit maakt het algoritme meer geschikt voor functiegeneratie, voor het bepalen van oplossingen van simultane vergelijkingen en voor optimalisatie.

Er werd voorts een volledig interactieve "Graphical Users Interface" voor het conceptuele motorontwerp ontwikkeld. Alle systeemcommando's zijn visueel toegankelijk door middel van een muis en besturingsknoppen. Iedere motorconfiguratie kan visueel gerepresenteerd worden door een netwerk van standaardcomponenten samen te stellen. Dit kan ook met een combinatie van eerder gedefinieerde submodellen. Het visuele netwerk kan daarna gebruikt worden om parameters op te vragen of te wijzigen.

Vanwege de vereiste software-prestaties en om praktische redenen werd CAGED ontwikkeld als een op zichzelf staand systeem. Alle software werd voor dit doel ontwikkeld en geschreven door de auteur.

Curriculum Vitae

The author was born on April 16, 1965 in Tehran, Iran. He completed his primary, secondary and first year of high school in Tehran before moving abroad, in June 1980, to continue his studies. A list of educational institutions attended and the degrees/certificates received is given below:

Sept.76-Jun.79: *Kharazmi Secondary School*, Tehran, Iran. Studied basic sciences, literature and arts. Sept.79-May.80: *Kharazmi High School*, Tehran, Iran. Studied exact sciences. Jun.80-Jun.81: *Boarzell Tutorial College*, Hurst Green, England. Basic orientation and learning the English language. Sept.81-Jun.82: *Hastings College of Arts & Technology*, Hastings, England. Received GCE 'O' Levels. Sept.82-Jun.83: *Bath Technical College*, Bath, England. Received GCE 'A' Level in Applied Mathematics. Sept.83-Jun.84: *Brunel Technical College*, Bristol, England. Received GCE 'A' Levels in Applied Mathematics, Pure Mathematics and Physics. Sept.84-Jun.88: *Loughborough University of Technology*, Transport Technology Department, Loughborough, Leics., England. Received Bachelor of Technology (*B.Tech.*) with honours in Aeronautical Engineering and Design. (*B.Tech.* = *B.S.* + 1 year integrated professional/industrial training). Sept.88-Jun.89: *University of Michigan*, Ann Arbor, USA. Aerospace Engineering, Gas Dynamics Division. *M.S.E. (Aerospace Engineering)* received with honours. Sept.89-Present: *Delft University of Technology*, Delft, The Netherlands. Aircraft Design and flight mechanics group, Aerospace Engineering Faculty. *Ph.D.* expected in Dec. 1994.

Since 1989, the author has been employed as *Junior Faculty Research* at the Aircraft Design and Flight Mechanics Group (vakgroep a2L), Faculty of Aerospace Engineering, Delft University of Technology.