



NLR-TR-2014-150

A new combustor and emission model for the gas turbine simulation program GSP

S.C.A. Kluitters, W.P.J. Visser and E.R. Rademaker

Nationaal Lucht- en Ruimtevaartlaboratorium

National Aerospace Laboratory NLR

Anthony Fokkerweg 2

P.O. Box 90502

1006 BM Amsterdam

The Netherlands

Telephone +31 (0)88 511 31 13

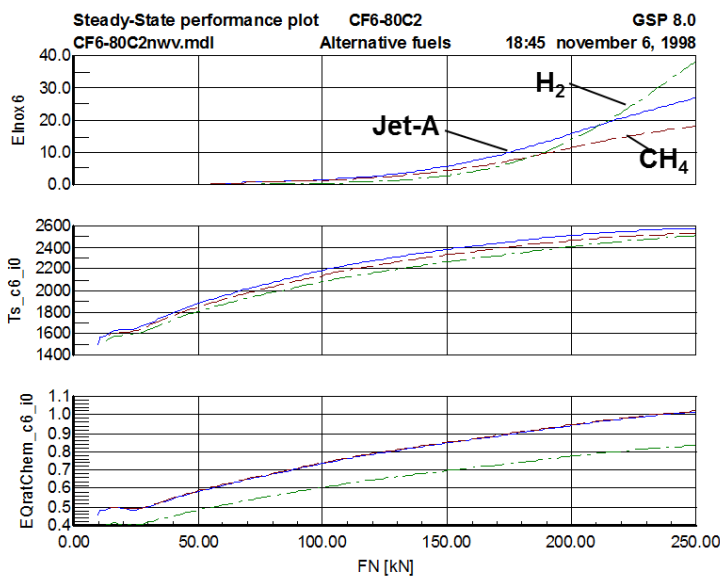
Fax +31 (0)88 511 32 10

www.nlr.nl



Executive summary

A new combustor and emission model for the gas turbine simulation program GSP



Effects of applying alternative fuels (methane and hydrogen) on NO_x

Problem area

With the increasing attention to gas turbine exhaust gas pollution, a need has emerged to quantify effects of a variety of operational variables such as engine condition, fuel type and water/steam injection on the emission levels. An effective approach to address this need is to integrate combustor emission models in gas turbine performance models. NLR's generic gas turbine performance simulation environment (GSP) has therefore been extended with a number of features for accurate analysis of these effects on the major exhaust

gas emissions NO_x, CO, UHC and smoke.

Description of work

The work has been carried out in the late nineties at NLR as a graduation assignment for the authors study Aerospace Engineering at the Technical University of Delft. His main supervisor was W.P.J. Visser.

S.C.A. Kluiters worked out a real thermo-chemical gas model based on NASA CEA. Main difference between the current and NASA model is that NASA minimizes the

Report no.

NLR-TR-2014-150

Author(s)

S.C.A. Kluiters
W.P.J. Visser
E.R. Rademaker

Report classification

UNCLASSIFIED

Date

April 2014

Knowledge area(s)

Gasturbinetechnologie

Descriptor(s)

Performance Simulation
Thermodynamic model
Emissions

so-called Gibbs Energy function, whereas Kluiters made use of the so-called equilibrium constants to calculate thermal equilibrium determined by gas composition and thermodynamic parameters as pressure and temperature.

Kluiters coded in Delphi-Pascal the gas model and successfully compared the obtained results with those of NASA CEA.

W.P.J. Visser has substantially contributed to the creation of the real gas model and has been responsible for the integration and implementation of the gas model in GSP and can therefore be regarded as co-author.

Results and conclusions

During the last decade the real gas model implemented in GSP has proved its value in many

applications varying from gas turbine performance predictions with flexible fuel handling (up to a user-specified fuel composition based on various species), life prediction of gas turbine components, adaptive modeling and engine emissions calculations.

Applicability

This report originally and unofficially had been published as a NLR Internal Memorandum (VH-98-010) with limited access. But due to its detailed description of the thermo-chemical gas model implemented in GSP and the relevance to GSP (for its further developments and applications), this official version has been published. This NLR Technical Report can be designated as a GSP reference document.



NLR-TR-2014-150

A new combustor and emission model for the gas turbine simulation program GSP


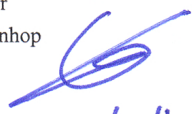

S.C.A. Kluiters¹, W.P.J. Visser¹ and E.R. Rademaker

¹ TU-Delft

No part of this report may be reproduced and/or disclosed, in any form or by any means without the prior written permission of NLR.

Customer	National Aerospace Laboratory NLR
Contract number	- - -
Owner	NLR
Division NLR	Aerospace Vehicles
Distribution	Limited
Classification of title	Unclassified
	April 2014

Approved by:

Author S.C.A. Kluiters W.P.J. Visser E.R. Rademaker  16/04/14	Reviewer O. Kogenhop  17/04/14	Managing department AVGS A.M. Vollebregt  22-4/14
Date: 11-04-2014	Date: 11-04-2014	Date: 11-04-2014

Preface

A previous version of this report was prepared at the National Aerospace Laboratory (NLR) in the Netherlands as a graduation assignment for the author's study Aerospace Engineering at Delft University of Technology. In a summarised form, it is also published as a paper called 'Modelling the Effects of Operating Conditions and Alternative Fuels on Gas Turbine Performance and Emissions', written by W.P.J. Visser and the author. This paper was presented at the RTO Applied Vehicle Technology Panel Symposium on Gas Turbine Engine Combustion, Emissions and Alternative Fuels, held 12th to 16th October in Lisbon, Portugal.

People who are primarily interested in a quick overview and some results could read the paper or the summary of this report. For readers not familiar with thermodynamics and emissions, it might be advisable to read the appendices on these subjects in advance. The author wishes to thank all the people who helped in the creation of this memorandum, especially W.P.J. Visser, for all the support, remarks and advises.

The current report originally has been published as in NLR internal memorandum (VH-98-010). But due to its detailed description of the thermo-chemical gas model implemented in GSP and the relevance as GSP reference document, it has been decided to publish it as an official NLR Technical Report. W.P.J. Visser has substantially contributed to the creation of the real gas model and has been responsible for the integration and implementation of the gas model in GSP and can therefore be regarded as co-author. E.R. Rademaker's role can be best described as editor. Changes with respect to the original report have been incorporated and are based on minor errors, GSP extensions and thoughts developed during the last decade.



This page is intentionally left blank.

Summary

With the increasing attention to gas turbine exhaust gas pollution, a need has emerged to quantify effects of a variety of operational variables such as engine condition, fuel type and water/steam injection on the emission levels. An effective approach to address this need is to integrate combustor emission models in gas turbine performance models. NLR's generic gas turbine performance simulation environment (GSP) has therefore been extended with a number of features for accurate analysis of these effects on the major exhaust gas emissions NO_x , CO, UHC and smoke.

First, possible gas turbine fuels were studied. Apart from the common gas turbine fuels like jet fuels and natural gas, a large number of alternative fuels emerge. This is not only because of desired exhaust gas emissions reductions, but it is also due to the predicted shortage of fossil fuel resources. The compositions of these fuels, especially of gasification products, can differ widely, which necessitates a flexible fuel specification interface for GSP.

Considering this flexible fuel specification, together with the wish to implement an improved combustor and emission model, led to the conclusion that the GSP gas model lacked both applicability and accuracy. Therefore, a new gas model was implemented, derived from the gas model used by the NASA CEA-program. In the new gas model, extended composition description and improved chemistry modelling have substantially increased the applicability and accuracy. The gains of this new gas model are also visible outside the combustion chamber, where evaporation/condensation and dissociation and their effect on component performance can be modelled now.

After implementation of the new gas model, combustor and emission models were studied. The choice was made to develop a new generic multi-reactor combustor model. The heat release and temperatures are calculated assuming chemical equilibrium. The emission formation is modelled applying simple kinetic schemes.

A combustion chamber model is built by dividing the combustion chamber liner volume into an array of reactors. In each reactor four flows can enter: the flow from the reactor before, an oxidant flow coming from outside the liner, the fuel flow, and a flow of water/steam. These four flows are assumed to mix instantaneously and reach chemical equilibrium at the reactor exit. In general, two types of emission formation are discerned: instantaneous formation in a flame and gradual formation throughout the combustion chamber. The instantaneously formed emissions are added to the total amount of emissions present so far. The gradual formation

determines emission formation rate equations (i.e. equations for time derivatives of the emissions). The (equilibrium) temperature, composition and the actual emission concentration at each reactor exit are used to calculate the emission formation rates. These formation rates are numerically integrated using the trapezium rule.

Four mechanisms of NO_x formation are modelled: prompt NO_x , fuel NO_x , thermal NO_x and NO_x formation by the N_2O mechanism. The amounts of prompt NO_x and fuel NO_x are calculated using empirical equations. They are supposed to be instantaneously formed in flames. Thermal NO_x and NO_x formed by the N_2O mechanism are assumed to be formed throughout the combustion chamber.

For CO emissions the assumption is made that the fuel instantaneously reacts to CO (and H_2O), and is subsequently (gradually) oxidised to CO_2 by one chemical reaction.

For UHC emissions, the fuel is converted to an amount of jet fuel or methane, depending on the fuel used. Both the jet fuel and methane are (partially) oxidised in the subsequent part of the combustion chamber.

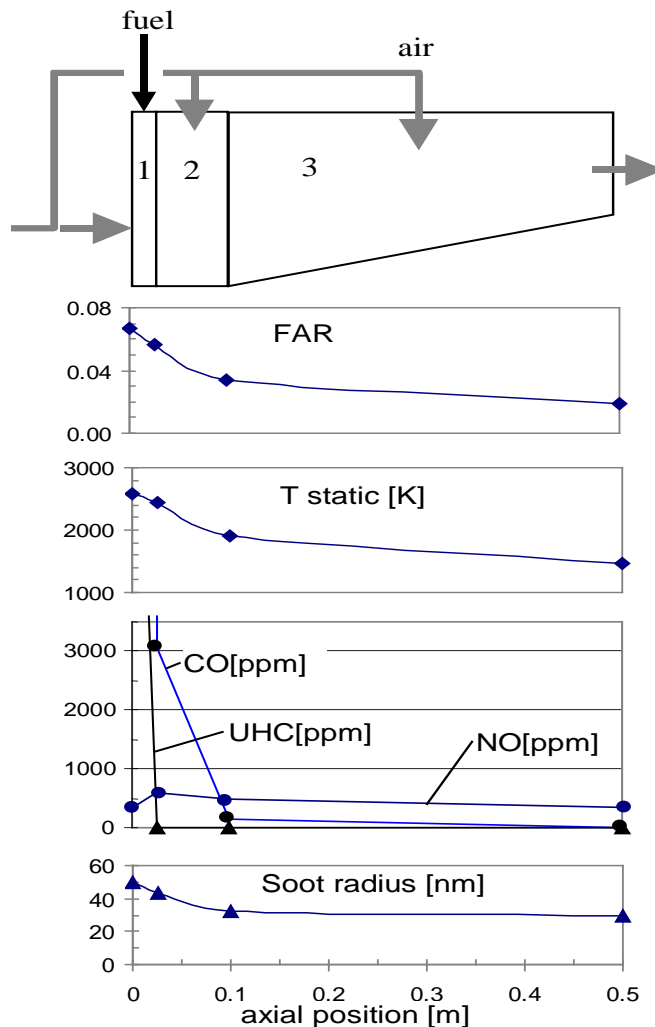
For smoke (soot), the assumption is made that the soot particles are spherical. An empirical equation is used to predict formation, while kinetic-type expressions are used to calculate the smoke oxidation.

The current model is preferably used as sensitivity analysis tool, i.e. to calculate effects on performance and emission parameters relative to reference values. Therefore, tuning to the reference values is necessary.

A demonstration of the combustor model is presented in the figure on the next page. In this figure, a multi-reactor model and design point results for the CF6-80C2 combustion chamber are shown. The combustion chamber is divided into three reactors. In the above two graphs, the fuel-air-ratio (FAR) and static temperature are shown as a function of the x-co-ordinate (which is parallel to the combustion chamber centre line). In the third graph and in the fourth graph, the emission mole fractions and the soot radius are shown as a function of the x-co-ordinate.

Validation of the emission models was performed using measurement data for the CF6-80C2. For this large turbofan engine, also the effects of non-standard ambient conditions, engine deterioration and application of alternative fuels were studied.

Another demonstration of the combustor model consisted of predicting effects of applying a low-calorific-value fuel in a gas turbine designed for natural gas on performance and emissions.



CF6-80C2 3-reactor model and design point results

Before further improvement of the combustor model, more validation studies are needed using detailed combustor data of a variety of engines and operating conditions.

Possible improvements are the implementation of an alternative (numerical) integration scheme, better modelling of the influence of the cooling (wall) flow on emissions and better modelling of the division of the liner cooling air over the reaction zones. The current generic multi-reactor combustor module can be used for easy implementation of improved emission models in the future.

Contents

1	Introduction	15
2	Gas turbine fuels and exhaust gas emissions	16
2.1	Fuels	16
2.1.1	Introduction	16
2.1.2	Liquid fuels	17
2.1.3	Gaseous fuels	18
2.2	Alternative fuel effects on performance	19
2.3	Exhaust gas emissions	20
2.3.1	Pollutants	20
2.3.2	Emission monitoring	22
2.3.3	Regulations	22
3	Gas models for gas turbine performance calculations	24
3.1	Constant specific heat gas model	24
3.2	Temperature dependent specific heat gas model	25
3.3	Variable composition gas model	26
3.4	Variable composition kinetic gas model	27
4	The new GSP 8.0 gas model	28
4.1	Choice of a new gas model	28
4.2	Description of the new gas model	29
4.2.1	General	29
4.2.2	Relations between thermodynamic and thermal transport properties	30
4.2.3	Modelling of composition changes	31
4.3	Application of the new gas model	33
4.3.1	Applying changing compositions	33
4.3.2	Determining compressor and turbine performance	33
4.3.3	Extensions in the user interface	34
5	Gas turbine combustor models	34
5.1	Combustion flow modelling	35
5.1.1	Black box model	35
5.1.2	Multi-reactor model	36
5.1.3	One-dimensional model	37

5.1.4	Multi-dimensional model	38
5.2	Combustion chemistry modelling	38
5.2.1	Flame sheet model	38
5.2.2	Chemical equilibrium model	39
5.2.3	Non-equilibrium and kinetic scheme chemistry	40
6	The new GSP 8.0 combustor model	40
6.1	Flow modelling	41
6.2	Chemistry modelling	42
6.2.1	Calculation of the equilibrium temperature and composition	43
6.2.2	Exhaust gas emissions modelling	44
7	Validation	58
7.1	The CF6-80C2 model	59
7.1.1	General	59
7.1.2	Emission predictions	60
7.2	The GE LM2500 model	66
7.2.1	General	66
7.2.2	Performance predictions	67
7.2.3	Emission predictions	70
7.3	Discussion on the results	71
8	Conclusions and Recommendations	72
	References	75
	Appendix A Graduation assignment	79
	Appendix B Combustion chemical reactions	80
	B.1 Reaction kinetics	80
	B.2 Dissociation	82
	B.3 Heats of reaction and heats of formation	82
	B.4 Calculating equilibrium compositions	85
	Appendix C Detailed description of gas models	87
	C.1 GSP 7.0	87
	C.2 GasTurb 7.0	90

C.3 GasTurb 8.0	92
C.4 NASA CEA-Program	93
C.4.1 Calculation of thermodynamic and thermal transport properties	93
C.4.2 Calculation of equilibrium compositions	99
C.5 A constant specific heat gas model	99
Appendix D Detailed description of combustor models	101
D.1 GSP 7.0 101	
D.2 GasTurb 7.0	102
D.3 GasTurb 8.0	102
Appendix E Formation of emissions	103
E.1 Formation of UHC, CO and Smoke	103
E.2 Formation of Nitrogen oxides	103
Appendix F NO_x-reductions in gas turbines	110
Appendix G An introductory description of GSP	112
G.1 A general introduction to GSP	112
G.2 GSP and the gas model	113
Appendix H Details of the new GSP 8.0 gas model	115
H.1 General remarks	115
H.2 Determination of thermodynamic and thermal transport properties as a function of temperature, pressure and composition	116
H.3 Modelling of composition changes	118
H.3.1 Solving the enthalpy balance	118
H.3.2 Determination of equilibrium compositions at a given temperature	121
H.4 Total versus static temperatures and pressures	123
Appendix I Calculation of compressor and turbine performance with real gas effects	125
I.1 Calculating outlet conditions	125
I.2 Using maps	126
Appendix J Calculating combustion equilibrium	130
J.1 Solving the enthalpy balance	130
J.2 Calculating equilibrium at a given temperature	134

Appendix K Delphi code of the gas and combustor model	137
K.1 Structure of the gas and combustor model	137
K.2 Pascal code of the new procedures	143

List of Symbols

A	: Area (m^2);
a	: Speed of sound (m/s);
C	: Concentration (kmol/m^3), (mol/cm^3);
c_p	: Specific heat at constant pressure (J/kg/K);
c_v	: Specific heat at constant volume (J/kg/K);
CEA	: (NASA) Computer program for calculation of complex chemical equilibrium compositions and applications;
CEM	: Continuous Emissions Monitoring;
CFD	: Computational Fluid Dynamics;
CO	: Carbon monoxide;
CO ₂	: Carbon dioxide;
Dp	: Mass of gaseous pollutant (g);
DUT	: Delft University of Technology;
E	: Energy (J);
EI	: Emission index (g/kg fuel);
FAR	: Fuel-air-ratio (-);
F ₀₀	: Rated (maximum) thrust (kN);
GE	: General Electric;
GSP	: Gas turbine Simulation Program;
H	: Hydrogen mass percentage (%), Heat (change) (J/kg);
h	: Enthalpy (J/kg);
ICAO	: International Civil Aviation Organisation;
IDLE	: Minimum thrust position of gas lever;
IGCC	: Integrated Gasification Combined Cycle plant;
ISA	: International Standard Atmosphere;
K _p	: Equilibrium constant using partial pressures;
L	: Characteristic length (m);
LHV	: Lower Heating Value (J/kg);
LPG	: Liquefied Petroleum Gas;
LTO	: Landing-takeoff cycle;
M	: Mach number (-), Molar mass (g/mole);
m	: Mass flow (kg/s), Mass fraction (-), Order of reaction (-);
N	: Total number of moles (-), Number of revolutions (rad/s);
n	: Number of moles (-), Total number of moles of gaseous species per gram mixture (mole/g);

NASA	: National Aeronautics and Space Administration;
NLR	: National Aerospace Laboratory;
NO_x	: Nitrogen oxides;
NG	: Number of Gases;
NS	: Number of Species;
P	: Power (W);
PEM	: Predictive Emissions Monitoring;
PR	: Pressure Ratio (-);
p	: (Total) Pressure (Pa), (bar), (atm);
ppm	: Parts per million;
Q	: Heat (J);
R	: Universal gas constant (=8314.51 (J/kmole/K));
RNI	: Reynolds Number Index (-);
R_g	: Specific gas constant (J/kg/K);
Re	: Reynolds number (-);
rpm	: revolutions per minute (/min);
S, s	: Entropy (J/kg/K), (J/mole/K);
SN	: Smoke number (-);
SO_x	: Sulphuric oxides;
T	: (Total) Temperature (K);
t	: Time (s);
UHC	: Unburned hydrocarbons;
V	: (Flow) Velocity (m/s);
W	: Work (J);
X, x	: Mole fraction (-);
γ	: Ratio of specific heats ($=c_p/c_v$) (-);
η	: Efficiency (-);
μ	: Dynamic viscosity (kg/(ms)), (μP);
Π_{00}	: Reference pressure ratio (-);
ρ	: Density (kg/m ³);
Φ	: Entropy function;
ϕ_{ij}	: Viscosity interaction coefficient between species i and j (-);
ϕ	: Equivalence ratio (-);
ω	: Specific surface oxidation rate (g/cm ² /s);

Subscripts

3	: compressor exit;
A	: activation;
a, air	: air;
c	: corrected;
comb	: combustion;
comp	: compression;
eq	: equilibrium;
F	: flow;
f	: combustion products in case of stoichiometric combustion, fuel;
g	: (gaseous) medium;
i, j	: specie i, j;
in	: inlet;
l	: air;
p	: (constant) pressure;
R	: rotor blade tips;
r	: reaction;
s	: virtual medium (used for calculation ease);
st	: standard conditions;
stoich	: stoichiometric.

1 Introduction

In the mid 1940's the first symptoms of smog were encountered in the Los Angeles area. It appeared that the increasing amounts of hydrocarbon fuels burned, caused by a growing industry and population, were responsible for the increased air pollution. Ever since, the growing demand for energy has resulted in higher amounts of pollutants emitted into the atmosphere. A part of the pollution is caused by gas turbines. Especially aircraft gas turbines are suspected to play an important role in ozone layer depletion and global warming, emitting pollutants higher in the atmosphere.

During the last few decades, the environmental problems have worsened, triggering extensive research by universities, manufacturers and research institutes on formation of pollutants, and on the effects these pollutants have on the atmosphere. The results of this research are used in an attempt to lower pollutant emissions and reduce their effect on living circumstances. On the manufacturers side reduced emissions are achieved by detailed modelling of the processes in the combustion chamber (using CFD). On the operational side, attempts to reduce emissions are performed by optimising operating conditions, such as engine condition, aircraft flight procedures, fuel type and water/steam injection. An effective approach to analyse operating conditions effects on emissions is to integrate emission models in gas turbine performance models, like NLR's Gas turbine Simulation Program (GSP).

So far, GSP has been primarily used to study operational aspects like diagnostics and failure analysis, where transient and static behaviour can be calculated under varying operating conditions. A few years ago, GSP was extended with a simple model predicting gas turbine emissions. However, the applicability of this model is limited.

The main purpose of the work described is to extend GSP in order to enable accurate predictions of the effects of applying different fuels as well as water/steam injection on gas turbine performance and emissions.

In the second chapter of this report, a general introduction to fuels, alternative fuel effects on performance and exhaust gas emissions is given. In the third chapter different types of gas models are studied. Chapter 4 contains the new gas model implemented in GSP 8.0 (the new version). In chapter 5 a number of combustor models are described, and the GSP 8.0 combustor model is discussed in chapter 6. Validation of the new gas and combustor model is presented in chapter 7. Chapter 8 contains the conclusions and recommendations.

2 Gas turbine fuels and exhaust gas emissions

The first paragraph of this chapter describes the most important categories of fuels to be used by gas turbines and the central theme of the second paragraph is gas turbine performance using different fuels. In the third paragraph a closer look is taken at exhaust gas emissions.

2.1 Fuels

2.1.1 Introduction

Common fuels used in gas turbines are hydrocarbons, obtained by distillation of crude oil. Since the 1973-1974 fuel crisis and the subsequent rise in fuel prices, the interest in diversification of fuels has grown. Diversification is also attractive because the world's fossil fuels resources are limited. In figure 2.1 (Hein, Ref. 17) the distribution of fossil energy resources in the world is shown and in figure 2.2 (Hein, Ref. 17) the accumulated world energy consumption. Because of the strongly growing demand for energy and the limited natural gas resources, it is to be expected that the end of the world's natural gas supplies will be reached in the next century.

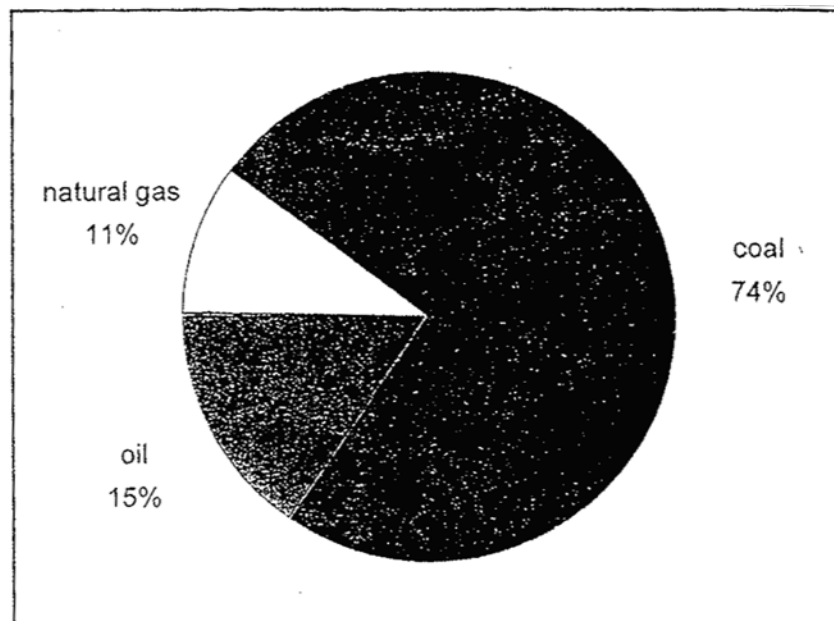


Figure 2.1 Distribution of fossil energy resources in the world

Alternative fuels are gasification products from coal or biomass (like straw). From figure 2.1 it is obvious that coal resources are present in large amounts. Because biomass grows on fields, it can be used eternally. Another important alternative fuel is hydrogen, although the production of hydrogen costs a lot of energy itself.

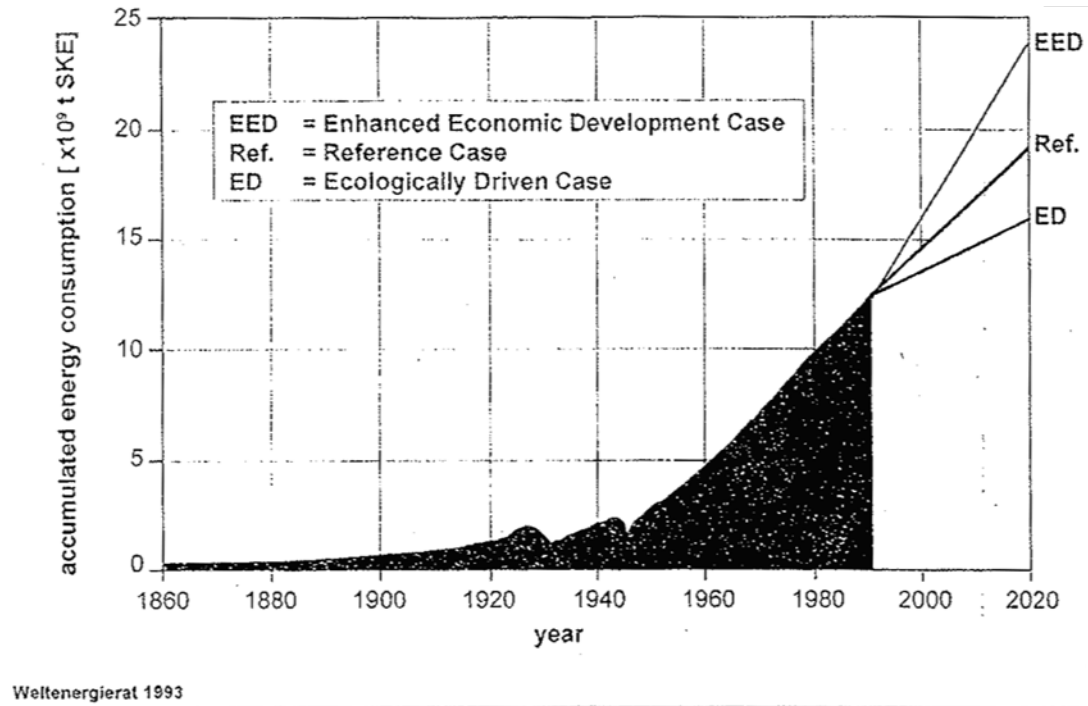


Figure 2.2 Accumulated world energy consumption

2.1.2 Liquid fuels

Gas turbines can be designed to use almost any liquid hydrocarbon fuel. The liquid fuels that are easiest to handle are fuels that are liquid at room temperature and atmospheric pressure. These are generally the hydrocarbons containing between four and twenty C atoms per molecule.

Important in this category are kerosenes, used for aircraft engines, and diesel.

Fuels that are gaseous at room temperature can be made liquid by keeping them under high pressure and/or low temperature. In the case of hydrocarbon fuels, examples are liquefied natural gas (primarily methane), or liquefied propane or butane. Hydrogen can also be used as a liquid fuel, but the temperature should then be kept as low as 33 (K) under high pressure or even lower if kept under lower pressures.

Although gas turbines are relatively tolerant concerning the liquid fuels to be used, the specific application of the gas turbine can inhibit the use of certain fuels. For example, aircraft fuels have to meet a number of stringent requirements, for reasons of safety and because of strongly varying ambient conditions. At low temperatures occurring during long-range high-altitude flights (temperatures as low as 230 (K) were recorded), the fuels should not freeze and must possess sufficient lubrication properties for movable parts within the fuel system. Also, excessive evaporation losses must be precluded and wide flammability limits are required for easy in-flight restarting of the engine. A high calorific value is desirable to enable longer ranges at the same fuel weight. These are only a few of the desired requirements.

Different requirements for jet fuels have led to typical fuel standards for military and civil aviation: kerosene-type fuels (Jet A-1, JP-5, JP-6, and JP-7) and wide-cut fuels (JP-4). These fuels differ in properties like flash and freezing point. They typically have a heating value of about 43 (MJ/kg) and an H/C-ratio between 1.9 and 2.0.

2.1.3 Gaseous fuels

Common gaseous fuels are shown in table 2.1, including their composition and lower heating value (Bokhorst, Ref. 4). Natural gas is a widely used fuel for gas turbines, but the application of other (low-calorific-value) fuels is considered. An example is the interest in application of (low-calorific-value) fuels, generated as a by-product of industrial processes (e.g. refinery waste gas).

Table 2.1 Composition and lower heating value of a number of gaseous fuels (Bokhorst, Ref. 4)

Fuel	Typical composition (in volume%)	Lower heating value	
		(MJ/m ³)	(MJ/kg)
Natural gas	75-95% CH ₄ , C ₂₊ , N ₂ , CO ₂	30-45	35-55
LPG (gaseous)	C ₃ H ₈ /C ₄ H ₁₀ -mixture	90-125	45-60
Refinery waste gas	40-90% H ₂ , CH ₄ , C ₂₊	15-40	50-100
Gasification gases:			
Air blown	10-15% H ₂ , 10-30% CO, 30-60% N ₂ , CO ₂	4-6	4-6
Oxygen blown	40-65% CO, 30-40% H ₂ , CH ₄ , CO ₂ , N ₂	8-15	8-15
Coke oven gas	50-55% H ₂ , 28-32% CH ₄ , C ₂₊ , CO, N ₂ , CO ₂	20-35	45-65
Blast furnace gas	55-60% N ₂ , 25-30% CO, CO ₂ , H ₂	3-4	3-4

An interesting process is making gaseous fuels from solid fuels, the so-called gasification. Gasification is achieved in a gasifier, e.g. a fluidised bed. In a fluidised bed, air or oxygen is guided through the solid fuel at a high temperature. In this way partial oxidation (in air or oxygen) of the fuel is achieved, as well as degassing. After the partial oxidation some combustible components are left. Gasification can also be in the form of pyrolysis of the fuel (i.e. breaking up the fuel molecules into smaller molecules), without using oxygen or air.

In table 2.1 it can be seen that gasification gases normally have a low calorific value, even in case of oxygen blown gasifiers. To increase the heating value, the gases could be mixed with a fuel having a higher heating value, like propane or butane.

In an Integrated Gasification Combined Cycle plant (IGCC) the gasifier and gas turbine are combined in one cycle. If a pressurised fluidised bed is used as gasifier, bleed air taken from the

compressor is used for the gasification process. The gasification gases are fed back to the gas turbine to serve as fuel. An interesting thing is that the partial oxidation has made the temperature of the gasification products quite high, so the energy content of the gasification products can differ quite a lot from the heating value at standard temperature. For more information on gasification, see (Hein, Ref. 17).

Coal gasification may be a good option for the future: according to figure 2.1 the coal resources are a lot bigger than the natural gas supplies. However, the coal supplies are also limited and burning of coal causes greenhouse gases in the atmosphere. Therefore, more permanent sources of energy are searched, not producing greenhouse gases. One possible source of energy for the future is biomass gasification. Examples of biomass are straw, wood and non-food cereals. An example of using biomass gasification products in a gas turbine designed for natural gas is given in chapter 7.

Gasification gases can contain a number of different species. Most species are light: propane and butane for example are hardly present in these fuels. As already said, gasification can be partial oxidation, so CO, CO₂ and H₂O can be present in quite large amounts. If the gasification is air blown, N₂ can also be present in quite large amounts.

2.2 Alternative fuel effects on performance

Applying alternative fuels in gas turbines can strongly influence gas turbine operation. Assuming that the mass flow at the turbine inlet is choked, the mass flow there can be calculated with the formula (Bokhorst, Ref. 4):

$$\frac{m\sqrt{T}}{p} = A^* \sqrt{M} \cdot \sqrt{\frac{\gamma}{R} \left(\frac{2}{\gamma+1} \right)^{\frac{\gamma+1}{\gamma-1}}} \quad (2.1)$$

where:

m	= mass flow (kg/s),
T	= (total) temperature (K),
p	= (total) pressure (Pa),
A^*	= critical flow area (m ²),
M	= molar mass (kg/mole),
γ	= ratio of specific heats (c_p/c_v) (-),
R	= universal gas constant (J/mole/K).

When fuels with different heating values are used to produce the same amount of power or thrust, the fuel mass flow changes, and for that reason the total mass flow also changes. The

terms on the right-hand side of the above equation (2.1) normally don't change very much with changing fuel composition. Therefore, the right hand side of equation (2.1) can be assumed constant.

If the left-hand side of equation (2.1) is constant, it is obvious that an increase (decrease) in mass flow necessitates an increase (decrease) in pressure and/or a decrease (increase) in temperature. A decreasing turbine inlet temperature could cause a lower efficiency or delivered power (or thrust). However, if the pressure increases, so will the compressor pressure ratio. The compressor pressure ratio has a direct effect on gas turbine efficiency in general and on compressor efficiency and stall margin in particular.

Using a low-calorific-value fuel, one needs a higher fuel mass flow to reach the same delivered power or thrust. If the turbine flow capacity remains constant, the turbine inlet temperature could decrease to balance the rise in mass flow. If the turbine inlet temperature doesn't decrease enough, a rise in pressure is necessary to balance the bigger mass flow, which also means a rise in compressor pressure ratio. This rise in compressor pressure ratio will make the stall margin smaller. If the stall margin becomes too small, a number of solutions can be applied. One is increasing the turbine flow capacity, another is increasing compressor load (e.g. by bleeding compressor air). This last option can be desirable when the gas turbine is part of an Integrated Gasification Combined Cycle (IGCC) with pressurised fluidised bed combustion: the bleed air can then be used for the gasification process: it will be fed back into the combustion chamber. Apart from the stall problems, also rotor over-speed problems can occur because of the mismatch between turbine and compressor. It appears that stall problems are more severe for single shaft engines (with fixed power turbine) while rotor over-speed problems are more severe for free power turbines (see chapter 7).

2.3 Exhaust gas emissions

2.3.1 Pollutants

In table 2.2, the pollutants emitted by gas turbines as well as their detrimental effects on health and environment are mentioned. As can be seen, the first five pollutants mentioned have direct effect on human health as well as on the environment. The greenhouse gas CO₂ has no direct effect on human health, but can cause global warming. It is believed that condensed water (H₂O) in the troposphere and the stratosphere also leads to global warming. In this condensation process an important role is played by smoke and sulphur emissions: due to these emissions, condensation nuclei are formed.

Table 2.2 Effects of pollutants on health and environment (Bruin, Ref. 7)

Pollutant	Effects on Health	Effects on environment
CO	Toxic	Tropospheric ozone production
UHC	Toxic	Global warming Tropospheric ozone production Smog
Smoke	Respiratory problems Cancer	Global warming
SO _x	Toxic	Global warming Corrosive Acid rain
NO _x	Toxic	Global warming Tropospheric ozone production Stratospheric ozone depletion Smog Acid rain
CO ₂		Global warming
H ₂ O		Global warming

The first gas turbines produced relatively large plumes of smoke. Apart from smoke, they also injected considerable amounts of carbon monoxide (CO) and unburned hydrocarbons (UHC) into the atmosphere. The production of these emissions gradually decreased as the combustion process became better understood and as higher turbine entry temperatures were reached, striving for higher efficiencies. These higher temperatures shifted the emissions production from CO, UHC and smoke to NO_x-production. A positive effect of the increase in overall efficiency, apart from lower fuel costs, is that the emissions of the greenhouse gases CO₂ and H₂O are lowered. Because sulphuric oxides are only produced if sulphur is present in the fuel, SO_x emissions can be reduced by prior sulphur removal from the fuel.

Nowadays, CO and UHC are primarily produced at IDLE power levels. Therefore they are only important in the vicinity of airports. Smoke and NO_x-emissions are primarily produced at high power settings (in take-off). NO_x-emissions are the most important exhaust gas emissions of today. Attempts to lower NO_x-production have primarily focussed on alternative combustion chamber design: a few alternative combustion chamber shapes are presented in appendix F.

2.3.2 Emission monitoring

Two methods of emissions monitoring are currently in use: Continuous Emissions Monitoring (CEM) and Predictive Emissions Monitoring (PEM) (Botros, Ref. 5). The main advantage of CEM is that pollutants are on line measured. Disadvantages are (among other things) the necessity to frequently calibrate the equipment, the fact that the technique is not truly continuous and the high costs of operation and maintenance. In contrast, PEM techniques offer greater accuracy and the possibility to provide advance results for specified operating conditions. For more information on emission monitoring, readers are referred to (Botros, Ref. 5).

Extensive work has been done on predicting (and reducing) emissions from gas turbines and a lot of models have been proposed. The majority of the models concern NO_x -formation. Often, these models are based on measurements, which can be performed using complete gas turbines or by using only the combustion chamber. In case of aircraft gas turbines, measurements can be performed on the ground in test cells, possibly simulating flight conditions at higher altitudes and in real flight (Jentink, Ref. 21).

The emissions can be quantified in a number of ways. A method often used is stating the fraction of the specie relative to the total of all species (as percentage or parts-per-million, abbreviated: ppm). Different fractions can be used, like volume fractions ($\text{ppm}(v)$) or mass fractions ($\text{ppm}(m)$). In GSP the emission index (EI) is used for UHC, CO and NO_x -production. The emission index is equal to the number of grams (g) of the pollutant per kilogram (kg) of fuel used. To quantify the smoke emissions, the smoke number (SN) is often used, a number between 0 and 100 denoting the level of staining of a special filter.

In chapters 5 and 6 more attention will be paid to combustion and emission modelling when the combustor model is discussed.

2.3.3 Regulations

Maximum gas turbine emissions are limited by legislation because of their negative impact on environment and human health. Legislation can differ from country to country.

In the case of aircraft gas turbines, maximum allowable emissions are set by the International Civil Aviation Organisation (ICAO). The ICAO certification resulted from the engine emission regulation program, started by the Environmental Protection Agency in the U.S. in 1970, mainly aimed at reducing emissions of UHC and smoke around airports. In 1981 the regulations were included in the engine certification program. These regulations are shown in table 2.3. The pollutants considered are CO, UHC, smoke, and NO_x -emissions. As can be seen, the emission

limits depend on the date of manufacture of the engine, the rated thrust (F_{00}), the reference pressure ratio (Π_{00}) and the application (subsonic/supersonic).

Table 2.3 ICAO regulatory levels for Dp/F_{00} or Dp/F_{00}^ (Bruin, Ref. 7)*

Engine Type	Subsonic turbojet/turbofan			Supersonic turbojet /turbofan
Date of Manufacture	$\geq 1-01-1983$	$\geq 1-01-1986$	$\geq 1-01-1996$	$\geq 18-02-1982$
Smoke (SN)	$83.6(F_{00})^{-0.274}$ or 50, whichever is higher			
Rated thrust	>26.7 (kN)			-
CO (g/kg)	-	19.6		$140(0.92)^{\Pi_{00}}$
UHC (g/kg)	-	118		$4550(\Pi_{00})^{-1.03}$
NO _x (g/kg)	-	$40+2\Pi_{00}$	$32+1.6\Pi_{00}$	$36+2.42\Pi_{00}$

F_{00}^* : Rated thrust when afterburning is used

So far, compliance with maximum emission limits is only checked at certification of engines, during a *landing and take-off (LTO) cycle*. This cycle includes the operations performed by aircraft as it descends from an altitude of 914 meter (3000 ft) on its approach path to the time it subsequently attains the same altitude after take-off: approach, idle, take-off and climb out. In these four situations exhaust gas emission measurements are performed. All the parameters are corrected to sea level ISA conditions using a relative humidity of 60%.

However, the major part of the emissions is emitted in cruise mode of the flight. Therefore, regulation on maximum emission limits during cruise-conditions is considered. Another option for the future is measuring and limiting emissions from engines during service-life, because the exhaust gas emissions can increase when engines become older.

For industrial gas turbines, the emissions of NO_x and SO_x are limited. There is no general limit on smoke, but local regulations must be observed. For automotive gas turbines, the same maximum emission levels apply as for automotive piston engines.

3 Gas models for gas turbine performance calculations

In order to implement a better model for the prediction of exhaust gas emissions, the current gas model of GSP (version 7.0) has to be improved. A gas model is defined here as a model that describes the thermodynamic and thermal transport properties of a (possibly two-phase) medium under changing circumstances. As the changing circumstances, in case of gas turbines, primarily changing temperatures and pressures are meant. When the temperature and/or pressure change, the composition of the medium can also change significantly. This change in composition also has to be modelled by the gas model.

For gas turbine applications, the most important thermodynamic properties to be determined are the specific heat at constant pressure (c_p), the enthalpy (h), the entropy (s), the ratio of specific heats (γ), the specific gas constant (R_g), and the speed of sound (a) as a function of composition, temperature and pressure. The only relevant thermal transport property for the moment is the dynamic viscosity (μ).

A lot of different gas models are possible. The models can range from very simple to very complex. If the gas model becomes more complex, more time will be needed for calculations, but the accuracy of the model will also increase, and the applicability will be wider. In this chapter, a few examples out of the possible range of gas models are shown. The first gas model is the most simple one and the last model is the most complex one. Examples of the gas models are described in appendix C.

3.1 Constant specific heat gas model

One of the most simple gas models thinkable is a model using constants for the specific heat at constant pressure, the ratio of the specific heats and the specific gas constant:

$$C_p = \text{Const.}$$

To make the model a little more generally applicable, several constant values can be used for the thermodynamic properties. Each constant value adheres to a certain gas stream. Gas streams can for example be air, combustion gases using fuel A and combustion gases using fuel B. The composition of a gas stream is assumed to be constant.

An example of a constant specific heat gas model is presented in paragraph C.5.

3.2 Temperature dependent specific heat gas model

If the number of constants used for the thermodynamic properties is more and more extended, using different values for different temperatures and pressures, the choice can be made to put the constants in tables and to interpolate between the constants. A limited description of the composition can be given by specifying characteristic ratios. Examples are the H/C-atom ratio and O/C-atom ratio or fuel-air-ratio. When the fuel-air-ratio is used, the amount of fuel added so far in the gas turbine is meant, not the actual amount of fuel present.

If this approach is used, the thermodynamic properties have become variables, depending on temperature, pressure and ratios specifying the composition. In case of the specific heat:

$$C_p = C_p(T, p, \text{ratios})$$

Usually, the pressure dependence of the specific heat is indirect. It is caused by the fact that pressure changes can involve composition changes (i.e. changes in the ratios). The physical meaning of the terms in this equation can be clear or rather obscure, depending on the ratios used. For example, if fuel-air-ratio is used, the equation often contains (among others) thermodynamic data for air (for the limiting case when the fuel-air-ratio is zero), which are easily imaginable. If H/C and O/C-ratios are used, the terms often have a more empirical character, which makes it less easy to understand the equations.

Increasing the accuracy of the models can be achieved by taking smaller steps in ratios, temperature and pressure, making the tables larger and larger. This will take more computer memory. Another way of achieving better predictions is by improving the interpolation. Linear interpolation can produce inaccurate results if the function is very non-linear. Better results are found by interpolating using curves. An example of using curves for interpolation is 'spline-interpolation' using parabolic lines, and logarithmic interpolation using logarithmic functions. Improving the interpolation process takes more computing time, but no extra memory. Another way of interpolating using curved lines is using the tables to make least-squares-polynomials. An advantage compared to the previously described interpolation is that this interpolation is only carried out once: when the polynomial is determined (not during the program calculations, but during programming). Using these polynomials, values for thermodynamic properties can be calculated easily. Generally speaking, with a higher degree polynomial, the results are more accurate.

Although the composition of the flows is described, composition changes due to changing temperatures and pressures (e.g. dissociation and evaporation) can only be described if appropriate ratios are chosen. The H/C and O/C-ratio or fuel-air-ratio are not appropriate,

because these ratio's only change if different flows are put together, not if dissociation or evaporation occurs. If for example the ratios like $\text{H}_2\text{O(g)}/\text{H}_2\text{O(l)}$ or CO_2/CO are used, the effects of dissociation and evaporation on thermodynamic properties can be modelled. However, the number of ratios needed will be quite large, so the question emerges whether it wouldn't be wiser to use a complete description of the medium in concentrations (or fractions) of all the species present. That approach is used in the next gas model to be described.

Examples of this type of gas model are the gas models used in GSP 7.0 and GasTurb 7.0 & 8.0. These models are described thoroughly in appendix C. In both models, the composition of the working medium in the gas turbine is determined from the fuel-air-ratio only, and important thermodynamic functions are calculated using the fuel-air-ratio and seventh degree polynomials, dependent on the temperature. The entropy is also a function of pressure. Only the GasTurb gas model calculates the dynamic viscosity.

An important difference between GasTurb 7.0 and GasTurb 8.0 is that version 8.0 offers users the possibility to choose from (a limited number of) different gas turbine fuels, where GasTurb 7.0 always assumes jet fuel. However, the approach used for the gas model is the same in GasTurb 7.0 and GasTurb 8.0. The fuel-air-ratio is used in both to account for variations in the composition.

3.3 Variable composition gas model

Although the temperature dependent specific heat gas model, described in paragraph 3.2, produced already more accurate results than the constant specific heat gas model, the applicability of this model was still rather limited, because of the trouble in describing composition changes. A logical extension of the model is describing the composition of the medium by the concentrations or fractions of the species present. Now, the large variety in compositions encountered in gas turbines can be described accurately and the different thermodynamic properties have become a function of the complete composition, pressure and temperature. For c_p :

$$C_p = C_p(T, p, eq.composition)$$

To find the values for thermodynamic variables for a mixture, the first step can be calculating the values for the individual species calculated. The values for the mixture are the weighted average of the values for individual species (see for example the gas model described in paragraph C.4). In this case, the terms in the above equation have a clear physical meaning.

Now that the influence of composition changes on thermodynamic properties can accurately be modelled, the question arises, how to model changes in composition. An easy way to do this is by including the possibility to calculate equilibrium compositions as well as frozen (=constant) compositions. The equilibrium composition is the composition that results when all the reactions have reached equilibrium: then the formation of all species is equal to their destruction.

An example of a variable composition gas model is present in ‘The Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications’ (CEA). A description of this program can be found in (Gordon, Ref. 15) and (McBride, Ref. 30). This program, developed by NASA, is capable of calculating rocket performance, Chapman-Jouguet detonation waves, shock tube parameters for both incident and reflected shocks and chemical equilibrium compositions. The chemical equilibrium compositions can be calculated when one of six combinations of two thermodynamic state functions is specified. This program contains a lot of thermodynamic data, valuable for large temperature ranges up to 20000 (K). In paragraph C.4 the gas model of the CEA-program is described.

3.4 Variable composition kinetic gas model

The modelling of composition changes in the variable composition gas model was greatly improved compared to the temperature dependent specific heat gas model by introducing the possibility to calculate equilibrium and frozen compositions. However, this is still a limited chemistry description.

A further improvement in the gas model can be achieved by taking into account other chemical situations than equilibrium and frozen. In this model, the thermodynamic functions remain (in general) functions of temperature, pressure and composition. For c_p :

$$C_p = C_p(T, p, \text{non} - \text{eq.composition})$$

These non-equilibrium compositions can in general be calculated in several ways. One is by adopting a kinetic scheme, and calculating the concentrations as a function of time using the reaction rate equations (see appendix B). This will, however, involve a large computational effort, because the ongoing reactions are coupled and will involve changes in temperatures and concentrations, thereby influencing reaction rates.

Several other ways are also proposed to predict non-equilibrium (and non-frozen) compositions. Keck (Ref. 24), for example, proposed a ‘Rate-controlled constrained-equilibrium theory’. Here, a sort of equilibrium is calculated, but this equilibrium is assumed to be constrained by kinetic schemes and possibly other constraints.

4 The new GSP 8.0 gas model

4.1 Choice of a new gas model

When looking at gas models, one should consider the applicability of the model as well as the fidelity (accuracy). The GSP 7.0 gas model is unfit for implementing a better emission model with variable fuel specification and two-phase flows, because it lacks both fidelity and applicability:

- The polynomials used to calculate thermodynamic properties as functions of temperature, pressure and fuel-air-ratio are only valid for temperatures lower than 1600 (K).
- Using (only) the fuel-air-ratio to determine compositions, it is not possible to model the effects of changing compositions due to (e.g.) evaporation or dissociation on gas properties.
- No chemistry model is present to find the compositions when evaporation or dissociation occurs.
- Using the fuel-air-ratio to determine compositions is not appropriate if a large flexibility in fuel compositions (especially to be expected for gasification products) is to be handled.

The chemistry model mentioned (in the third point) is not used for combustion and emission calculations, because the combustion chamber has its own chemistry modelling. However, because of increased amounts of water due to water/steam injection and because of the high turbine inlet temperatures encountered in today's high-pressure gas turbines, evaporation and dissociation are also likely to occur outside the combustion chamber. Therefore, evaporation and dissociation effects should also be modelled in other components than the combustion chamber.

Because of the second and fourth mentioned shortcoming, the fuel-air-ratio has to be abandoned, and the third point necessitates the implementation of (improved) chemistry modelling. When implementing a new gas model, attention has to be paid to validity at higher temperatures.

Instead of the fuel-air-ratio, other ratios could be applied to describe the composition. However, in chapter three it has already been noticed that a number of ratios are necessary to describe the composition if evaporation and dissociation can occur. The choice was made to describe the composition using fractions of the species present. This more generic approach enables easy extending of the gas model when additional species need to be added (e.g. because of exotic fuels). Another advantage of this approach compared to ratios is that the composition in species is easily imaginable and provides better insight into the conditions of the working medium.

As stated above, chemistry modelling is necessary to determine the composition when evaporation or dissociation occur. The choice was made to enable calculating equilibrium (and frozen) compositions. Although it is questionable whether the equilibrium will be reached, it does give the opportunity to model evaporation and dissociation with limited calculation effort.

4.2 Description of the new gas model

4.2.1 General

It is obvious, that the chosen model is a variable composition gas model, as described in paragraph 3.3. In this new gas model, the gas model from the NASA CEA-program is used to a large extent. The main difference is that the equilibrium compositions are calculated in a different way.

In the new gas model, the composition is specified in mass fractions of the species. The mass fraction is used because the mass flow is used to specify the working medium flow (and not the volume flow or the number of moles). Also, the relevant thermodynamic (output) data are all expressed per kg and not per mole or m^3 . Moreover, the volume of liquid water is assumed to be negligible, so the water present wouldn't appear in a composition in volume fractions. Where needed, the mass fractions can easily be changed into volume or mole fractions. The vector containing the species possibly present is: (CO_2 , CO , O_2 , Ar , $\text{H}_2\text{O}(\text{g})$, $\text{H}_2\text{O}(\text{l})$, H_2 , CH_4 , C_2H_6 , C_2H_4 , C_3H_8 , C_4H_{10} , O , H , OH , NO , N_2O , C_xH_y , N_2).

Polynomials are used instead of tables, because the polynomials are readily present (in the NASA CEA-program), and they are assumed to be a good trade-off between calculation power, memory and accuracy. Polynomials can easily describe non-linear functions using only limited memory for the coefficients and using only limited computer calculation time. When applying tables for accurate description of non-linear functions, either the step size must be chosen small, resulting in large memory needed, or difficult interpolation is needed, resulting in long calculation times. An advantage of using the NASA data is NASA's good reputation and the fact that the data can easily be found in NASA reference papers, which are available in a lot of places. Because the NASA polynomials are valid for a limited temperature range, two sets of coefficients are used: one for the polynomial valid from 200 to 1000 (K) and one for the polynomial valid from 1000 to 6000 (K). With this gas model, the thermodynamic properties can accurately be found for temperatures high enough.

4.2.2 Relations between thermodynamic and thermal transport properties

The following thermodynamic and thermal transport properties are calculated as a function of temperature, pressure and composition:

- the specific heat at constant pressure (c_p),
- the enthalpy (h),
- the specific gas constant (R_g),
- the entropy (s),
- the ratio of specific heats (γ),
- the speed of sound (a),
- the dynamic viscosity (μ).

The calculation of these properties is to a large extent done in the same way as in the NASA CEA program. Details are shown in appendix H. Here, a few remarks are made.

The specific heat at constant pressure calculated is only valid if the composition remains constant during changes in temperature (frozen composition). If this is not the case, the composition change will involve a heat effect and a different specific heat from the one calculated above will result. Because the calculated specific heat depends on composition changes, it also depends on the chemistry description applied. Therefore, the NASA CEA-program not only calculates the frozen composition specific heat but also the equilibrium composition specific heat. NASA calculates the difference between these two specific heats, the $c_{p,r}$ (r for reaction), in the iteration procedure used to find the equilibrium composition. Because GSP doesn't use the same iteration procedure as NASA, in GSP only the frozen specific heat is calculated.

The slope of the enthalpy as a function of temperature is fixed: it is equal to the specific heat at constant pressure. The absolute value for the enthalpy can be chosen. Here, it is determined by the fact that the enthalpy of a specie at $T = 298.15$ (K) is equal to its heat of formation at $T = 298.15$ (K), like in the NASA CEA-program. This is adopted in order to keep the same coefficients for thermodynamic properties as NASA. As explained in paragraph H.3, it can prove handy in solving the enthalpy balance.

The ratio of specific heats is calculated in the same way as in the GSP 7.0 gas model. For gases, this is correct as long as the composition doesn't change because of dissociation. If the composition does change it is assumed to be a good approximation. For mixtures containing liquid water, the values found for γ correlate well with results found by using NASA's CEA-program.

4.2.3 Modelling of composition changes

Like the NASA gas model, the GSP 8.0 gas model has a limited description of kinetics, only using frozen or equilibrium compositions. If the chemical processes are assumed to be slow in comparison with the residence times in the components, a frozen composition is assumed. Otherwise, the composition will change and the equilibrium composition is calculated. Because the gas model only calculates the thermodynamic and thermal transport properties for a flow under changing temperatures and pressures, composition changes due to mixing of separate flows (and subsequent reactions) are not modelled by the gas model. Therefore, the most important composition changes to be modelled are due to evaporation (/condensation) or dissociation. Because evaporation and dissociation involve conversion between potential energy and heat, evaporation and dissociation also influence temperature itself. In order to find the composition and temperature when evaporation or dissociation occurs, the assumption is made that there is no heat loss (i.e. adiabatic reactions) or pressure loss during the evaporation and dissociation. The temperature and composition are then found in an iteration procedure solving the equilibrium composition belonging to a certain temperature and checking the enthalpy balance (also called energy equation) to see whether the temperature is correct. For an extensive description of checking the enthalpy balance and for the calculation procedures used to find the equilibrium composition at a given temperature, appendix H should be consulted. Here some general remarks are made on modelling of evaporation (/condensation) and modelling of dissociation.

Modelling of evaporation/dissociation

Usually, the only liquid (apart from fuel) present in the gas turbine will be water. This water can be water entering the inlet because of rain or because of test programs, to see whether flame out will not occur when lots of water enters the gas turbine. In old aircraft gas turbines, in the take-off water was inserted at the entrance of the compressor, to achieve higher thrust. Another possibility is that water is injected into the combustion chamber as a NO_x -abatement procedure. Young (Ref. 45) studied low-pressure steam turbines and discerns two kinds of non-equilibrium: thermal non-equilibrium and inertial non-equilibrium. Thermal non-equilibrium is caused by a temperature difference between droplets and surrounding gases and the fact that the droplets and water vapour are not in equilibrium. Inertial non-equilibrium is caused by a difference in speed between the droplets and the surrounding gases. Young concludes that for applications with small droplets (in the order of 0.1 to 1 μm) in low-pressure steam turbines, inertial non-equilibrium is negligible in comparison with thermal non-equilibrium. Although the conditions (including droplet size) in gas turbine components might be quite different, from conditions in low-pressure steam turbines, inertial non-equilibrium is neglected in the new gas model.

Young presents a model to describe thermal non-equilibrium, using geometric data and residence times. Because the number of geometric data for the gas turbine models used by GSP is to be minimised, the model described by Young is not used.

In the new version of GSP no thermal non-equilibrium is accounted for except for the difference between frozen and equilibrium compositions. In the inlet, the composition is supposed to be frozen, because of the very short residence time in the inlet, because the temperatures there are usually not very high, and because water droplets can be quite big. In all other components the equilibrium between liquid and gaseous water is supposed to be reached, because temperatures are usually higher than in the inlet. Also, in compressors and turbines the water droplets will be dispersed by the rotors and stators. This will result in smaller droplets and shorter times needed to evaporate.

Modelling of dissociation

Usually, dissociation doesn't occur at temperatures lower than 1800 (K). In case temperatures in the gas turbine (outside the combustion chamber) are higher than 1800 (K), dissociation is assumed to take place, and a new equilibrium is calculated. Because of the high temperatures, all liquid water is made vapour immediately. Because, on the other side, the temperatures are assumed to remain limited (below 2200 (K)) outside of the combustion chamber, only dissociation of carbon dioxide and water, forming carbon monoxide, hydrogen and oxygen, is assumed to be important. In other words, only the following reactions are assumed to be relevant:



As explained more extensively in appendix H, a system of five equations has to be solved to find the five unknown fractions determining the equilibrium composition at a given temperature. Solving this system is achieved by guessing one fraction, calculating the other fractions for this guess, as well as a new value for this fraction. The deviation between the new value and the guess is used to find the correct value for the guessed fraction (then the guess is equal to the newly calculated value) and therefore the correct equilibrium composition. Because the guessed fraction must be a non-zero fraction, and because of calculation advantages, the O_2 -fraction is used for this purpose. However, because of the limited accuracy of (standard) Delphi calculations, this will give problems if the O_2 -fractions becomes smaller than $1 \cdot 10^{-15}$. Normally, this will not be a problem, because this only happens in (very) rich mixtures, especially at low temperatures, which are not likely to occur outside the combustion chamber.

4.3 Application of the new gas model

The higher accuracy and more extended (variable) composition description of the GSP 8.0 gas model enable better (component) performance predictions, and more information for the users. The applicability of GSP 8.0 in general has also increased.

4.3.1 Applying changing compositions

Although evaporation and dissociation usually only occur in the combustion chamber, in GSP 8.0 effects of evaporation and dissociation in other components are also modelled. At the outlet of each component, two checks are performed. The first one is whether water (liquid or in gaseous form) is present. If this is the case a verification is made that the equilibrium between liquid and gaseous water is correct for the temperature and pressure at the component outlet (except for the inlet of course, because here the composition is frozen). If not, (adiabatic) evaporation or condensation is assumed to take place, changing the temperature and the composition as described above (in paragraph 4.2.3).

The second check is a temperature check. If the temperature exceeds 1800 (K), dissociation becomes relevant, and the (new) equilibrium composition and temperature at the component outlet are calculated.

Because the checks are only performed at the outlet, a small error may be introduced. In reality, the composition changes continuously and not with a step at the outlet of the component. In the compressor, for example, liquid water is assumed not to be compressed. In the current approach, the quantity of liquid water in the compressor is assumed to stay the same: only at the outlet of the compressor, evaporation is assumed to take place. Because in reality evaporation occurs throughout the compressor, more gaseous (and less liquid) water will be present in reality than GSP assumes. This extra amount of water vapour also has to be compressed. Thus, GSP underestimates the power needed to compress the medium in the compressor. Changing the zero-dimensional approach of the components in a one- (or more-) dimensional approach can solve this problem. However, this bears a lot of consequences, like the importance of the geometry of the components.

4.3.2 Determining compressor and turbine performance

Now that two-phase flows can occur, a decision has to be made on calculation of power needed to compress the liquids. The choice has been made to neglect power needed to compress the liquids (except fuel) in all the components. Also no power is gained when the droplets expand. The influence of Reynolds number on compressor and turbine performance is accounted for in GSP 8.0. This was made possible by adding the dynamic viscosity to the thermodynamic and

transport properties that are calculated in the new gas model. Therefore, the following dimensionless parameters are used to determine compressor and turbine performance:

- Isentropic efficiency,
- Pressure ratio,
- Corrected mass flow,
- Corrected number of rotations,
- Reynolds number index.

Because of the more complete composition description, the calculation of these parameters has changed. For additional data on the meaning of these numbers and the way of calculating them, readers are referred to appendix I.

4.3.3 Extensions in the user interface

The improved gas model provides users with more information. In all the components, the number of possible output parameters has increased. The composition of the working medium can be generated as output data in the inlet and outlet of all components. Also, the values of specific heat at constant pressure, enthalpy and entropy can be shown as output parameters in all inlets and outlets of components.

A change in the inlet is that humidity of air can be simulated now. This can even be rain or lots of water thrown into the gas turbine for flame out tests. The water can be specified as relative humidity (not higher than 100%), as volume percentage (water in gaseous form only because the volume of liquid water is assumed to be negligible) or as mass percentage of the airflow. Application of water injection into the compressor, done in earlier days to increase thrust in take-off of aircraft, can be simulated by assuming the water to enter the gas turbine at the inlet. The results obtained then are of course not exact.

5 Gas turbine combustor models

The gas models described in the previous chapters predict thermodynamic and thermal transport properties of a given medium under changing temperatures and pressures, also when the composition changes. However, at certain places in the gas turbine flows with different compositions are put together. In case of mixing, usually no reactions occur and the new composition can easily be found by taking the (mass flow) weighted average of the (mass) composition of the flows coming together. The temperature can be found by solving the enthalpy balance, as described in appendix H. In combustion processes, however, chemical

reactions occur on a big scale and a special combustor model is needed to provide accurate modelling.

Another reason to pay special attention to combustion modelling is because the exhaust gas emissions are (almost) completely formed here and the main target of the work described in this report is improving GSP's capability of predicting exhaust gas emissions, using different fuels and steam/water injection.

One of the difficult things about combustion modelling is that there is a strong interaction between chemical reactions and aerodynamics: chemical reactions usually have a big heat release influencing the flow field, while the flow field determines whether species come together and have the opportunity to react.

In this chapter, firstly a number of approaches for combustion flow modelling are presented. After that chemistry models are shown. Combustor models comprise a combination of a chemistry model and a flow model. Optimal yields would be achieved if more complex chemistry models would be combined with more complex flow models. In this way, the gain (e.g. better and more information) from applying a more complex model can be used by the other model. Because of large calculation times, however, combination of both a complex chemistry and flow model is unattractive for GSP.

5.1 Combustion flow modelling

5.1.1 Black box model

The first and most simple approach is modelling the combustion chamber as a (zero-dimensional) black box, like the other gas turbine components in GSP. In this black box, no model is applied to describe the mixing process: mixing is assumed to be infinitely fast and complete. All properties in the volume are assumed to be uniform. Consequently, no spatial distributions (e.g. of flow properties) can be calculated and the only possible variations are time-dependent transient variations. Effects like heat transfer to the walls can be taken into account.

This zero-dimensional box is called a well-stirred reactor. Only two places in the well-stirred reactor are important: the inlet and the outlet. Therefore, this flow model doesn't give information about the flows during the combustion process. This model can only be a reasonable approximation if there are no big temperature gradients and if the flow is uniform enough.

In programs for gas turbine modelling, like GSP and GasTurb, this approach is often used. In appendix D the combustor models of GSP 7.0 and GasTurb 7.0 & 8.0 are described.

5.1.2 Multi-reactor model

Compared to the black-box approximation, the combustor model can be extended by dividing the combustion chamber in a number of reactors: these models are so-called multi-reactor models (or multi-zone models). These reactors can be placed in an array but also parallel. If the reactors are zero-dimensional (as described in the last paragraph), they are called well-stirred reactors, and if they are one-dimensional, they are called 'plug flow reactors'. Also multi-dimensional reactors can be used, although multi-dimensional modelling is only common practice for CFD (see paragraph 5.1.4). If big temperature gradients occur, these places can be used as the border between two reactors. In this way well-stirred reactors can be used, even though temperature gradients occur.

If a multi-reactor model is used, it is easy to find a favourable trade-off between accurate modelling and not using too much time, because different types of reactors can be combined. A multi-dimensional reactor can be used, accompanied by a finer discretisation in those places of the combustion chamber where the processes occurring are more complex. The most important combustion zones in a diffusion-flame combustion chamber are shown in figure 5.1.

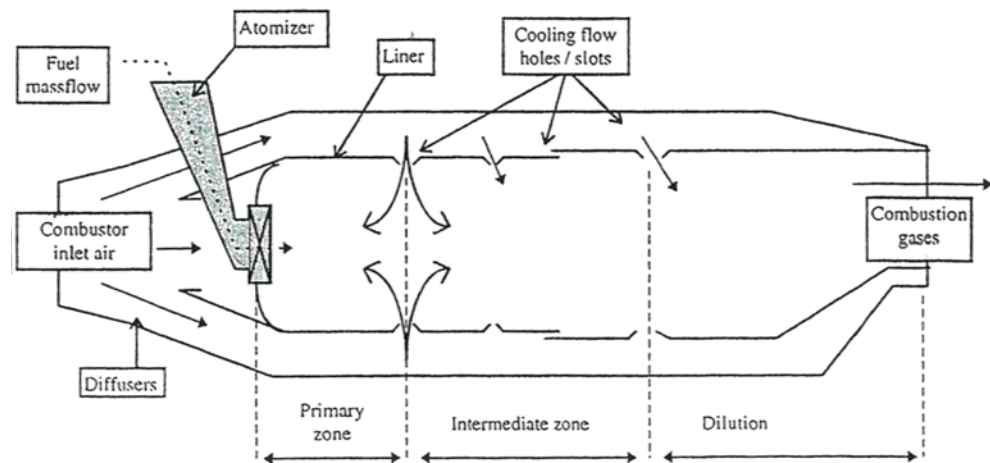


Figure 5.1 Main zones of a gas turbine combustor (Bruin, Ref. 7)

Examples of multi-reactor models are shown in figures F.1, F.2 and F.3 (in appendix F). These figures show a big advantage of applying multi-reactor models: different types of combustion chambers can be modelled using only simple combustion chamber models for different zones. In this way, the necessary computing time can remain limited.

5.1.3 One-dimensional model

An improvement in the combustion flow model compared to the well-stirred reactor can be achieved by admitting variations in the axial direction. When implementing this one-dimensional approach, one must ‘slice’ or ‘discretise’ in the (axial) x-direction, as shown in figure 5.2. In each volume that is formed by slicing, the mass, impulse and energy balance are applied. If chemical reactions occur, conservation of atoms must also be accounted for. Finite-rate mixing can be modelled by application of macro and micro mixing models.

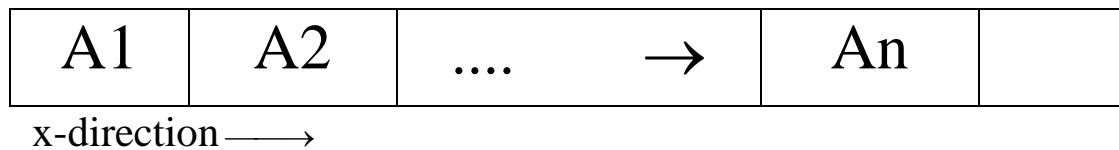


Figure 5.2 Slicing in the x-direction

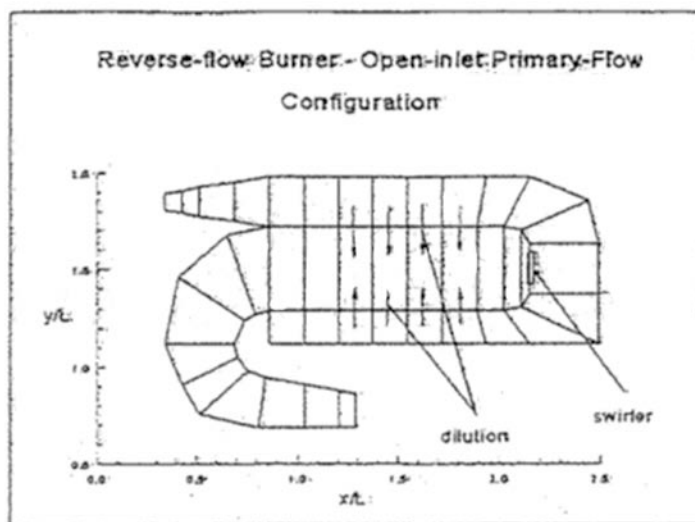


Figure 5.3 Discretisation of a reverse-flow combustion chamber

A problem using this approach is that the solution must be found simultaneously for all the segments, while the number of equations involved is quite large. Therefore, usually an iterative procedure is necessary.

An example of a one-dimensional combustor model is the unsteady, finite-rate model (Rodriguez, Ref. 38), proposed by Rodriguez and O'Brien. In figure 5.3, taken from (Rodriguez, Ref. 38), the discretisation of a reverse-flow combustor is shown. The space within the liner and the space around the liner are both sub-divided in discrete elements.

5.1.4 Multi-dimensional model

Further improving the flow modelling in the combustion chamber can be achieved by admitting variations, and therefore also discretising, in more than one direction: multi-dimensional models. These models are often referred to as CFD (Computational Fluid Dynamics) models, although, strictly speaking, CFD is a much broader term, also including for example one-dimensional models.

Multi-dimensional models are the most realistic models. In these models, a complete spatial distribution of flow properties (e.g. temperature) and composition can be found. Because of the large number of equations and volume elements involved, this approach takes a lot of computing time and memory space on even the fastest computers. Therefore, it can be advisable to reduce the number of dimensions if possible. Axisymmetric shapes, like the shape of a combustion chamber for example, can be modelled quasi two-dimensionally. These models get extremely complex if an accurate description of turbulence is required. Modelling of turbulent reactant flows is discussed (in Dutch) in (Peeters, Ref. 35). This book also contains a number of useful references.

5.2 Combustion chemistry modelling

5.2.1 Flame sheet model

The easiest way of modelling combustion chemistry is by assuming a one-step combustion reaction (a so-called flame sheet model). In case of a larger than stoichiometric amount of air this can be a reasonable approximation, given the fact that combustion reactions are usually quite fast, while conversion from reactants to products is almost complete. A necessary condition is, however, that the combustion temperatures are not too high, so that dissociation does not play an important role.

Although a one-step combustion reaction is an easy and sometimes reasonable approximation of the combustion process, it is not a very realistic approach, because an actual combustion process involves a great deal of reaction steps. Also, the information given by the model is very limited: no concentrations of radicals are calculated and only one temperature can be found: the final temperature at the burner exit assuming no dissociation.

Obviously, this model is inadequate in providing a reasonable description of emission formation and depletion: unburned hydrocarbons, carbon monoxide and smoke are not assumed to be present because of the one-step combustion reaction. Nitrogen oxide formation depends strongly on the temperatures encountered in the different combustor zones and therefore on the combustion progress. Therefore, the only way of predicting emissions while applying a flame

sheet model is by using empirical equations. These are found by combining measurement data with relevant gas turbine data (temperatures and pressures), and are usually not widely applicable.

In GSP 7.0 a flame sheet model is used for the combustion chemistry. Consequently, the emissions are predicted using empirical equations. Because GSP is primarily used to calculate off-design effects (see appendix G), the emissions in the design point of the gas turbine are usually known, and the empirical models can be tuned to the design values in order to enable more accurate emission predictions. The relevant changes in operation parameters (e.g. temperatures) compared to the design point are calculated in the form of ratios and the difference in emissions compared to the design point is predicted. These models are called ratio models (Bruin, Ref. 7). The main drawback of these models is that their applicability is limited to new (or revised) gas turbines working under standard conditions. In other words, they can't be used (for example) for gas turbines using alternative fuels or for gas turbines with deteriorated components.

5.2.2 Chemical equilibrium model

An improvement in combustion description can be achieved by applying equilibrium chemistry. As explained in appendix B, the equilibrium state is the state that the process relaxes to if it is given enough time: the concentrations of combustion products then remain unchanged as a function of time. Given the fact that combustion processes are usually rather fast, the equilibrium can be assumed to be reached.

Clear advantages compared to the flame sheet model are that fuel-rich combustion can be modelled now and that more information is available, like (equilibrium) radical species concentrations. Dissociation, likely to occur at the high temperatures encountered in combustion, can also be modelled now.

Although this model provides a better description of the combustion process it is by itself still inadequate in predicting exhaust gas emissions: incomplete burning (and subsequent formation of carbon monoxide, unburned hydrocarbons and smoke) of the fuel can't be modelled. Also, as the temperature of the gases decreases, the equilibrium model will predict the formed carbon monoxide and nitrogen oxide to react to nitrogen and carbon dioxide, if enough oxygen is present. In reality, the reactions that form carbon dioxide and nitrogen form carbon monoxide and nitrogen oxides 'freeze' (i.e. become very slow) at low temperatures and the carbon monoxide and nitrogen oxides remain present in the exhaust gases.

The combustion chemistry description in GasTurb 7.0 and 8.0 assumes equilibrium to be reached in the combustion chamber. However, this is only used to calculate the burner exit temperature. GasTurb does not include an emission prediction model.

5.2.3 Non-equilibrium and kinetic scheme chemistry

A further improvement compared to the equilibrium model can be achieved by applying a better description of chemical kinetics. This can be done in a number of ways. One can assume partial equilibrium: some reactions are assumed to reach equilibrium, others are not. For this last category of reactions, a (limited) kinetic scheme can be used. As already mentioned in the most complex gas model described in chapter three, Keck (Ref. 24) has described a way of calculating a constrained equilibrium.

Another option is making no prior assumptions about equilibrium and only using a kinetic scheme. However, each kinetic scheme can only be used for a limited range of combustion conditions (e.g. for equivalence ratios between certain values). In general, bigger schemes yield better results, but use more computing power. These bigger schemes are only available for a small number of fuels. The smaller the number of reactions get, the more limited the range becomes where the kinetic scheme can be applied. Also, there is still a lot of disagreement about rate constants: quantitatively correct results are difficult to achieve. Large kinetic schemes are not easy to use in gas turbine combustion chamber modelling, because of the very long computation time needed to solve the equations (simultaneously).

When this better modelling of chemical kinetics is applied, a reasonable calculation of emissions can be achieved: the formation and depletion of emissions can be calculated in the kinetic scheme. For an example of kinetic schemes involving NO_x formation and depletion, readers are referred to (Miller, Ref. 32). In case of a partial equilibrium, the general combustion can be assumed to be in equilibrium, while emission formation can be described by a kinetic scheme. One must bear in mind, however, that emission predictions with a deviation of 20 to 30% from the measured values, achieved using CFD, are very good results for the moment.

6 The new GSP 8.0 combustor model

In GSP 7.0, the combustion chamber is modelled as a black box. The chemistry is described by a one-step combustion reaction, not taking into account dissociation. Because of this limited chemistry description, empirical equations are used to predict the emission indices of pollutants. Because of the off-design character of GSP, the emission levels are known in the design point.

Therefore, the model present in GSP 7.0 is an empirical ratio model. Because only jet fuel can be selected as gas turbine fuel in GSP 7.0 and because the emission indices are known in the design point, the model suits the needs.

GSP 8.0, however, must have the possibility to predict the effect of applying different fuels and steam/water-injection on exhaust gas emissions. Therefore, the chemistry and flow modelling are improved. In the new combustor model, an approach is used similar to (Bozza, Ref. 6). First the combustion flow model is described and after that the chemistry model, where special attention is paid to emissions modelling.

6.1 Flow modelling

The GSP 8.0 combustor model is a multi-reactor model: the combustion chamber is divided into a variable number of zones (reactors). Only the processes within the liner are simulated, and the combustion chamber modelling starts at the (first) flame front. This means that mixing or vaporising prior to combustion is assumed to play a negligible role in the combustion (and emission formation) process.

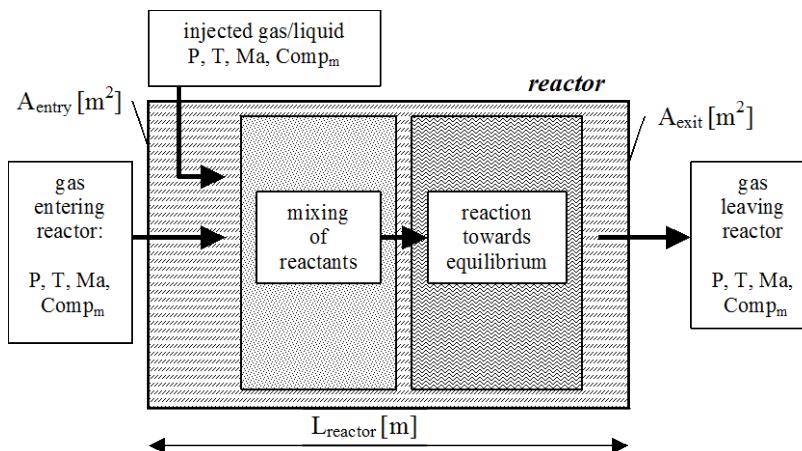


Figure 6.1 Schematic of a reactor

A general picture of a reactor is shown in figure 6.1. The reactors are of the well-stirred type, assuming infinitely fast mixing and uniform properties for each reactor. In each reactor, four flows can enter: the flow from the reactor before and three flows (in figure 6.1 denoted by one arrow) newly entering the liner of the combustion chamber: a fuel flow, a flow of inert species (e.g. water or steam) and a flow of the oxidant, usually air, coming from outside the liner.

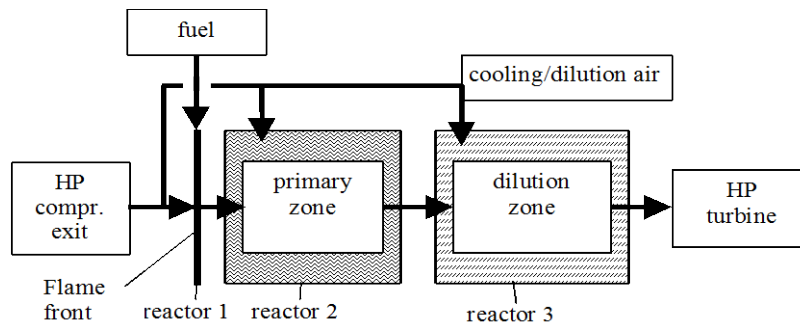


Figure 6.2 Simple example of a multi-reactor arrangement

A multi-reactor model is an array of reactors. The first reactor is the (first) flame front, that is modelled as a plane. The number of reactors is variable and can be chosen by users. Applying more reactors can improve the accuracy, but also results in longer computation times. No attempt is made to model the distribution of the cooling air outside the liner over the different zones inside the liner. To do this accurately would necessitate the use of CFD, which is considered a too heavy burden for GSP. The user specifies the flow distribution of the cooling flow in the design point, which can be reasonably estimated, and this distribution is kept constant under off-design operating conditions. This constant flow distribution may cause degradation of the results for operating points largely deviating from the design point.

In figure 6.2 an example of modelling a diffusion flame combustion chamber model is shown. The combustion chamber is divided into three zones: the flame front, primary zone and dilution zone. All the fuel is added in the flame front, while the flow leaving the compressor is divided over all three reactors.

6.2 Chemistry modelling

The chemistry modelling is a mix between equilibrium chemistry and kinetic schemes. The combustion of the fuel, determining the heat release and the temperature, is assumed to reach equilibrium. This can be justified by the fact that hydrocarbon reactions are generally rapid reactions (see for example Sturgess, Ref. 41). Justification for this assumption can also be found in (Hammond, Ref. 16), who compared application of kinetic schemes with an equilibrium model. Conclusions were that for the exact determination of the composition, an equilibrium model could not be used, but the completeness of combustion and the temperature could be fairly well approximated with the equilibrium assumption.

In emission modelling limited kinetic schemes are used. The kinetic schemes use the temperature and (equilibrium) radical concentrations, determined by the combustion equilibrium. By doing this, it is assumed that the emission formation itself doesn't influence the temperature. Normally, this is a good approximation because exhaust gas emission

concentrations are small. Although the radical concentrations are not equal to their equilibrium values (Hammond, Ref. 16), the equilibrium radical concentrations are used anyway because other radical concentrations (determined by kinetics) are not available and because the deviations from equilibrium are assumed to be small. The assumption is based on the fact that at the high temperatures that are usually encountered in gas turbines, the combustion process occurs in a short reaction zone and that the residence time in this reaction zone is rather short compared to the total residence time in the combustion chamber. Therefore, in the largest part of the combustion chamber, equilibrium is (almost) reached.

6.2.1 Calculation of the equilibrium temperature and composition

In every reactor, the assumption is made that the equilibrium state is reached at the exit surface (see figure 6.1). The way of finding the equilibrium temperature is essentially the same as the one applied in the gas model (see chapter 4 and appendix H) to find the temperature after dissociation or evaporation.

The most important difference is that the static temperature is used to find the composition, while the total temperature is used to solve the enthalpy balance. The reason for this is that the chemistry (rates of formation and depletion of species) is determined by mutual, relative speeds of molecules (a measure for the number and strength of collisions), and not by the speed the molecules all have in common. However, the movement of all the molecules together does affect the heat contained in the flow. Appendix H contains more information on static and total temperatures and pressures.

Details on the calculation of the equilibrium composition at a given temperature are given in appendix J. Solving the enthalpy balance is explained in appendix H. Here, some remarks are made on calculation of the equilibrium composition at a given temperature.

Determination of the equilibrium composition at a given temperature

The procedure used to find the equilibrium composition at a given temperature operates essentially in the same way as the procedure calculating equilibrium compositions when dissociation occurs, used in the gas model. Only now, the temperatures encountered can be higher, and consequently, additional species will appear due to dissociation. Therefore, the concentration of species O, H and OH are also calculated now. Also, NO and N₂O equilibrium concentrations are calculated now (these concentrations are used by the emission model, to be described further on), adding N₂ to the list of species whose concentrations are to be determined. Because of the (usually) high combustion temperatures all water is assumed to be vapour.

Like the determination of the equilibrium composition at a given temperature in the case of dissociation, described in paragraph 4.2.3, here also a system of equations has to be solved. The same approach is used here. The O₂-fraction is guessed, and for this guess, the other fractions are calculated. Also a new value for the O₂-fraction results from the other fractions. The deviation between the O₂-fraction guess and the new value for the O₂-fraction is used to find the O₂-fraction for which the guess and the newly calculated value are equal. Then, the equilibrium composition is found.

Like in the determination of dissociation equilibrium, a problem arises, when the O₂-fraction becomes smaller than $1 \cdot 10^{-15}$: then the fractions can't be correctly calculated anymore, because of limited accuracy of the (standard) Delphi calculations. O₂-fractions this low are only likely to be encountered in fuel-rich areas, especially when temperatures are low. Outside the combustion chamber fuel-rich zones usually don't occur. However, within the combustion chamber fuel-rich zones can be present. Usually, the equivalence ratios needed to find equilibrium O₂-fractions lower than $1 \cdot 10^{-15}$ are above (approximately) 1.7. Because these equivalence ratios are not likely to be found in combustion chambers, the equilibrium composition at a given temperature can usually be found.

6.2.2 Exhaust gas emissions modelling

6.2.2.1 General calculation procedure

The general calculation procedure applied is the same for all the exhaust gas emissions. Two types of emission formation (/depletion) processes are discerned: infinitely fast reactions and slower reactions. Infinitely fast reactions only take place in flames, in other words, in reactors where fuel is injected. They produce a step-wise rise in the emission levels. The slower reactions take place in all the reactors, as long as the conditions are favourable. They determine the rates of emission formation, i.e. the time-derivatives of the exhaust gas emissions. The exhaust gas emission levels are found by adding emission contributions from infinitely fast reactions and by numerically integrating the emission formation rates throughout the combustion chamber, from one zone exit plane to the other.

The numerical integration is carried out using the trapezium rule, also called the method of Crank-Nicolson (Kan, Ref. 23):

$$\begin{bmatrix} NO \\ CO \\ UHC \\ SN \end{bmatrix} (t_1) = \begin{bmatrix} NO \\ CO \\ UHC \\ SN \end{bmatrix} (t_0) + \begin{bmatrix} NO \\ CO \\ UHC \\ SN \end{bmatrix}_{step} + \left\{ \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \\ SN \end{bmatrix} (t_1) + \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \\ SN \end{bmatrix} (t_0) \right\} \frac{\Delta t}{2} \quad (6.1).$$

In this formula, Δt is equal to $(t_1 - t_0)$. The time derivative of the emission vector is a function of the emission vector itself. Therefore, this is an implicit method, and equation (6.1) has to be solved iteratively. Although this iterative procedure takes more calculation time than not implicit methods, it is preferred for GSP, because it is always stable, independent of the step-size used. This is very attractive for GSP because the step-size is equal to the reactor residence time, and therefore depends on the number of reactors. In this way, a GSP user can choose a number of reactors satisfying the required accuracy without having to worry about numerical stability.

Equation (6.1) is the original equation given by Kluiters and solely based on changes of the concentrations of species caused by chemical reactions. However, the values of the concentrations of the pollutants at the exit plane of the combustor are the results of two processes: the mixture of hot combustor flow (within the liner) with cooling flows (through the liner dilution holes) and the chemical reactions for non-equilibrium pollutant concentrations. The addition of the mixing effect of flows leads for the pollutant concentrations to a modified numerical integration method compared to (6.1) and is given by

$$V_1 \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_1) = V_0 \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_0) + V_0 \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix}_{step} + \left\{ V_1 \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_1) + V_0 \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_0) \right\} \frac{\Delta t}{2} \quad (6.1a).$$

V_0 and V_1 are representative gas volumes (before and after mixing) at both time steps. The ratio of these volumes is given by $\frac{V_0}{V_1} = \frac{W_0 T_0 P_1}{W_1 T_1 P_0}$ (6.1.b). The modified trapezium rules assuming that

$P_1 \approx P_0$ (the pressure in the combustor is nearly constant) becomes

$$\begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_1) = \frac{W_0 T_0}{W_1 T_1} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_0) + \frac{W_0 T_0}{W_1 T_1} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix}_{step} + \left\{ \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_1) + \frac{W_0 T_0}{W_1 T_1} \frac{d}{dt} \begin{bmatrix} NO \\ CO \\ UHC \end{bmatrix} (t_0) \right\} \frac{\Delta t}{2} \quad (6.1.c)$$

The soot diameter is exiled from this equation since the soot radius and the total number of soot particles remain constant by mixing (in contrast to the concentrations).

If more combustion chambers are used in the gas turbine, the NO_x concentration exiting the combustion chamber before is used as the starting concentration. Smoke, carbon monoxide and

unburned hydrocarbons are assumed to be burnt up in subsequent combustion chambers without influencing new formation or depletion of these emissions.

6.2.2.2 Equations for NO_x-emissions

The approach used in NO_x prediction is the same as in (Bozza, Ref. 6), (Fletcher, Ref. 12) and (Barrère, Ref. 3), although some prediction formulas and chemical reactions are added where thought necessary. As noted in appendix E, NO_x in the combustion chamber is the sum of NO and NO₂. In GSP 8.0's emission formation calculations, the (molar) NO_x emissions are calculated as if they consisted only of NO.

It is to be expected, that this will not introduce serious errors, because amounts of NO₂ formed in combustion processes are small (see e.g. Bokhorst, Ref. 4). NO₂ is encountered in two regions of the combustion chamber: in the (low temperature part of the) flame zone and the dilution zone. Within the flame zone NO₂ temporarily exists. It is formed by reaction (E.1). At higher temperatures, NO₂ reacts back to NO by reactions (E.2) and (E.3). NO₂ can only remain present as a result of chilling effects of cold fluid elements. Because of the recirculation in the primary zone of the combustion chamber, this amount of NO₂ will probably remain small. Formation of NO₂ from NO by reaction (E.4) in the dilution zone usually remains limited, because the temperatures are too high.

Besides, because the amount of NO_x is the sum of NO and NO₂ it doesn't matter for the amount of NO_x (when given in *moles*) whether it is formed by NO or by NO₂. The only mistake that is introduced comes from the facts that NO₂ will not participate in the same reactions as NO. However, the *conversion* of the (molar) amount of NO_x exiting the combustion chamber *to an emission index* is done as though the NO_x consisted of NO₂. This is recommended by ICAO and is also the standard in emission measurements.

In the new combustor model, four significant NO_x pathways are modelled:

- Prompt NO_x,
- Fuel NO_x,
- Thermal NO_x,
- N₂O mechanism.

Because the formation of prompt NO_x and fuel NO_x is usually rapid, both mechanisms to a large extent involving radicals that are only present in the main fuel reaction zone, fuel NO_x and prompt NO_x are supposed to be formed instantaneously. The amounts of prompt NO_x and fuel NO_x formed are calculated using equations found in literature and added to the amount of NO_x formed so far. Of course, prompt NO_x and fuel NO_x are only formed in reactors where fuel is

burned. Thermal NO_x and NO_x formation by the N_2O mechanism are assumed to be less rapid than prompt and fuel NO_x formation. They are integrated throughout the combustion chamber.

Prompt NO_x

The equation used to calculate the amount of prompt NO_x is taken from (Toof, Ref. 42):

$$[\text{NO}]_{\text{prompt}} = X_{\text{CH}} f(\phi) \sqrt{p} [\text{NO}]_{\text{eq, stoich}} \quad (6.2),$$

where:

- X_{CH} = mole fraction of hydrocarbon species in fuel (-),
- $f(\phi)$ = a function of the equivalence ratio ϕ (-),
- p = pressure (bar).

The square brackets denote mole fractions (here). In the equation it can easily be seen, that only hydrocarbon containing fuels are assumed to form prompt NO_x . This means, that prompt NO_x formation due to the accelerated Zeldovich mechanism and the N_2O mechanism (see appendix E) is neglected. This is a good approximation because in flames O and OH concentrations with values above equilibrium are usually only present at temperatures too low to enable the Zeldovich mechanism to form significant amounts of prompt NO_x (Miller, Ref. 32). The contribution of the N_2O mechanism to prompt NO_x formation is also discarded because it is only important at conditions where total NO_x emissions are low (Glassman, Ref. 14). Indeed, measurements described in literature only show significant amounts of prompt NO_x in hydrocarbon flames. For the pressure term, the static pressure is used. The most right term on the right-hand side of equation (6.2) denotes the NO concentration formed under stoichiometric equilibrium conditions.

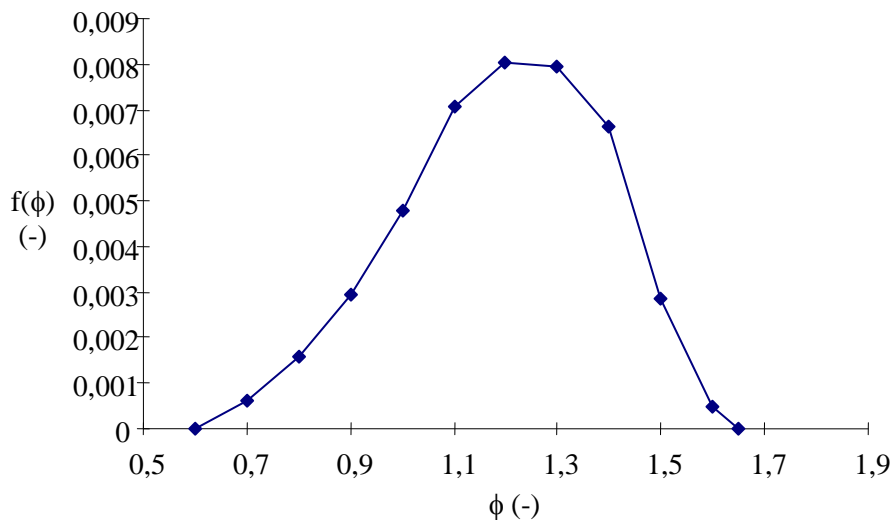


Figure 6.3 The function $f(\phi)$ from equation (6.2)

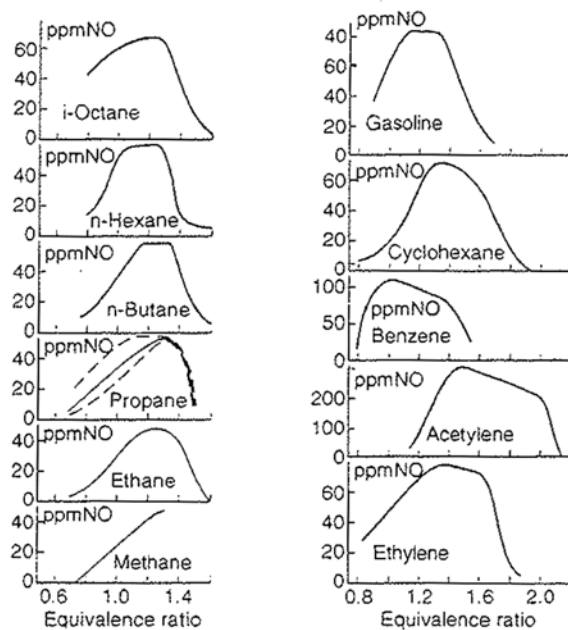


Figure 6.4 Prompt NO_x formation as a function of equivalence ratio for a number of hydrocarbons (Bachmaier, Ref. 2)

The function $f(\phi)$, shown in figure 6.3, is determined by the author using measurement data described in (Bachmaier, Ref. 2) and depicted in figure 6.4. From this article, it is obvious, that most hydrocarbons have negligible prompt NO_x formation at equivalence ratios below 0.6 and above 1.65. However, of the hydrocarbons used in the measurements, cyclohexane, ethylene (C_2H_4), acetylene (C_2H_2) and, to a lesser extent, gasoline did have significant prompt NO_x formation at equivalence ratios above 1.65. Because hydrocarbons like methane, ethane, propane and butane don't produce prompt NO_x at these equivalence ratios, the prompt NO_x formation above equivalence ratios of 1.65 is discarded, although this might introduce errors when applying gasoline or fuels with significant amounts of ethylene and acetylene.

An uncertainty in equation (6.2) is the pressure dependence, especially at high pressures encountered in aircraft and aero-derivative gas turbines. The pressure dependent terms in the above equation are the two most right terms on the right hand side of the equation. It can clearly be seen that a difference in pressures of a factor 40 results in a difference in prompt NO_x of a factor 6.3, if differences in the most right term are discarded. It is questionable whether this difference is also accounted in reality. No prompt NO_x measurements have been found in the high-pressure regions to validate the square root dependence of pressure. Validations will have to point out whether this assumption is a good approximation.

Fuel NO_x

In predicting the amounts of fuel NO_x, the important quantity to be determined is the conversion fraction, i.e. the fraction of total fuel bound nitrogen that is actually converted to NO_x. This conversion fraction depends strongly on the local combustion environment (e.g. equivalence ratio and fuel composition), but not on the way in which the nitrogen is chemically bound in the fuel (see e.g. Glassman, Ref. 14).

In literature, experimentally determined conversion fractions are given for a few low-heating value combustion processes (Kelsall, Ref. 25; Sato, Ref. 40 and Nakata, Ref. 34) and for ethylene flames (Fenimore, Ref. 11). These studies show that the conversion fraction can differ widely and that there are many different factors influencing it. For the moment, no effort is made in predicting the conversion fraction; the user can specify a constant conversion fraction. This conversion fraction can be changed for each new group of working points. Because the amount of fuel-bound-nitrogen is a fuel property, it is also user-specified.

Thermal NO_x

For the thermal NO_x formation, the extended Zeldovich mechanism is applied:



N₂O mechanism

For the nitrous oxides (=N₂O)-mechanism, almost all the reactions mentioned in appendix E are used. Only the last one is omitted, because it is assumed to have a negligible effect. The reactions are:



In reaction (6.6), M is a non-reacting collision partner (also called third-body specie) for N₂O. It is only important at low pressures, where reaction (6.6) behaves like a second order reaction (see appendix B); at high pressures, (6.6) behaves like a first-order reaction, so that M is omitted. Between the high-pressure and low-pressure case, there is a gradual change between first order and second order reaction, described by the so-called Lindemann fall-off rate (see

Glassman, Ref. 14). However, for simplicity, here only the limiting cases are described. The high-pressure case is used for pressures higher than 10 (bar) and the low-pressure case for lower pressures. For this reaction, N_2 is assumed to be the collision partner.

Using these reaction mechanisms for both thermal NO_x and the N_2O -mechanism, an equation for the NO_x formation rate is derived in the same way as done by Bozza (Ref. 6), Barrère (Ref. 3) and Fletcher (Ref. 12), only with a slightly different reaction mechanism. In this derivation, all the species are assumed to be in equilibrium, except for NO , N_2O and N . N and N_2O are assumed to be in steady state (not a function of time), but not in equilibrium. This last assumption is taken from (Lavoie, Ref. 27). For N , the steady state approximation is also found to be valid by (Botros, Ref. 5).

In the derivation, the so-called ‘one-way equilibrium reaction rates’ (Fletcher, Ref. 12) are used. These rates, denoted by R_i ’s, are the product of the forward specific reaction rate constant and the equilibrium concentrations on the left side of reaction equation ‘i’:

$$R_i = k_f \prod_{j=1}^{j=n} [X_j]_{eq} \quad (6.12).$$

For the forward specific reaction rate constant k_f , Arrhenius’ law (see appendix B) is used:

$$k = AT^\beta e^{\left(\frac{E_a}{RT}\right)} \quad (6.13).$$

Because the equilibrium concentrations are used, this product is also equal to the backward reaction rate multiplied with the equilibrium concentrations on the right side of reaction equation ‘i’. Further information on one-way equilibrium rates is provided in paragraph B.1. Using reactions (6.3) to (6.11), the following equations are found for the time derivatives of NO , N and N_2O :

$$\begin{aligned} \frac{d[NO]}{dt} = & R_{6.3} + \beta(R_{6.4} + R_{6.5}) + \gamma(2R_{6.8} + R_{6.10} + R_{6.11}) \\ & - \alpha(\beta R_{6.3} + R_{6.4} + R_{6.5} + 2\alpha R_{6.8} + R_{6.10} + R_{6.11}) \end{aligned} \quad (6.14),$$

$$\frac{d[N]}{dt} = R_{6.3} + \alpha(R_{6.4} + R_{6.5}) - \beta(\alpha R_{6.3} + R_{6.4} + R_{6.5}) \quad (6.15),$$

$$\frac{d[N_2O]}{dt} = R_{6.6} + R_{6.7} + R_{6.9} + \alpha(\alpha R_{6.8} + R_{6.10} + R_{6.11}) - \gamma(R_{6.6} + R_{6.7} + R_{6.8} + R_{6.9} + R_{6.10} + R_{6.11}) \quad (6.16),$$

$$\text{where: } \alpha = \frac{[NO]}{[NO]_{eq}} \quad (6.17),$$

$$\beta = \frac{[N]}{[N]_{eq}} \quad (6.18),$$

$$\gamma = \frac{[N_2O]}{[N_2O]_{eq}} \quad (6.19).$$

Because of the steady state assumption for N and N₂O, the left-hand sides of equations (6.15) and (6.16) are zero, and β and γ are found as a function of α and the relevant one-way equilibrium reaction rates. After substitution of β and γ in equation (6.14), the following equation is found for the NO formation rate:

$$\frac{d[NO]}{dt} = 2(1 - \alpha^2) \left\{ \frac{R_{6.3}}{1 + \alpha \frac{R_{6.3}}{R_{6.4} + R_{6.5}}} + \frac{R_{6.8} + \frac{R_{6.10} + R_{6.11}}{2(1 + \alpha)} \left(1 + \frac{\alpha R_{6.8}}{R_{6.6} + R_{6.7} + R_{6.9}} \right)}{1 + \frac{R_{6.8} + R_{6.10} + R_{6.11}}{R_{6.6} + R_{6.7} + R_{6.9}}} \right\} \quad (6.20).$$

This equation gives the NO formation rate, which is assumed to be equal to the NO_x formation rate, as a function of the temperature, the NO concentration and a number of equilibrium concentrations. It is obvious that the left-hand term between the curly brackets is thermal NO_x formation, while the right-hand term is due to the N₂O mechanism. Because the one-way equilibrium reaction rates can be calculated using the forward reaction rate with the appropriate equilibrium concentrations as well as the backward reaction rate with the adhering equilibrium concentrations, the number of equilibrium concentrations to be calculated can be minimised. In this case, only the NO and N₂O equilibrium concentrations are calculated, and not the N, NCO and NH equilibrium concentrations.

6.2.2.3 Equations for other emissions

The other three pollutant levels predicted are for carbon monoxide (CO), unburned hydrocarbons (UHC) and smoke.

Carbon monoxide emissions

In calculating carbon monoxide emissions, the assumption is made that the fuel reacts in a one-step infinitely fast reaction to carbon monoxide and water:



This reaction is only used for emission calculations; the temperature is calculated using the combustion equilibrium assumption. The reaction is assumed to take place in every reactor where fuel is inserted. The CO formed is supposed to be oxidised to CO₂ in the subsequent reactors of the combustion chamber. The assumption that the fuel oxidation to CO is quickly achieved in the thin fuel reaction zone, while the slower subsequent oxidation of CO to CO₂ is achieved in the zones after, is often made in literature, see for example (Sturgess, Ref. 41; Hammond, Ref. 16 and Westenberg, Ref. 44).

In the literature, a number of CO oxidation reactions are given, but most of the authors agree that the following reaction is the dominant reaction:



The H and OH radicals are formed by reactions involving the water formed by equation (6.21). Assuming that this is the only CO removing (or forming) reaction, together with the assumption that the O and OH concentrations are equal to the equilibrium values and together with the conservation of carbon atoms:

$$([CO] + [CO_2]) = ([CO] + [CO_2])_{equilibrium} \quad (6.23),$$

the following equation can be found for the rate of carbon monoxide removal (Chleboun, Ref. 8):

$$\frac{d[CO]}{dt} = -k_{6.22f}[OH]_{eq} \left\{ 1 + \frac{[CO]_{eq}}{[CO_2]_{eq}} \right\} ([CO] - [CO]_{eq}) \quad (6.24).$$

In this equation, the $k_{6.22f}$ is the specific forward reaction rate constant of equation (6.22). The equation is integrated through the combustion chamber zones like the equation for NO_x formation (6.20).

This equation is able to model the effect of rapid reaction down towards equilibrium CO concentration (because of the minus sign and the most right term on the right hand side) at relatively high temperatures and also to simulate the effect of frozen high CO concentrations

due to sudden quenching. In the latter case, the specific reaction rate constant will suddenly decrease to a very low value, thereby preventing further rapid CO oxidation.

Because of the large integration steps used and the high CO oxidation rates in the flames, equation (6.24) could very well predict very low CO concentrations. Because these very low concentrations are not realistic, the CO formation rate calculated using equation (6.24) is only used if the calculated CO concentration is higher than the equilibrium concentration. In other cases, the equilibrium CO concentration is used.

Unburned hydrocarbon emissions

To predict the emission levels of unburned hydrocarbons, again a global one-step hydrocarbon oxidation reaction is used, like (6.21). The possible hydrocarbon fuels to be selected (see appendix J) are divided into two categories. The first category contains jet fuels and diesel. These fuels generally contain large molecules. According to NASA (McBride, Ref. 30), the average properties of Jet-A can be described by assuming that it exists of $C_{12}H_{23}$ -molecules. When a jet fuel or diesel is chosen as fuel, the assumption is made that they consist of $C_{12}H_{23}$, and that they react according to the following equation (Pratt, Ref. 36):



In (Pratt, Ref. 36), the following equation is given for the rate of $C_{12}H_{23}$ -consumption:

$$\frac{d[C_{12}H_{23}]}{dt} = -10^{11.5} \left(\frac{p}{p_0} \right)^{-0.815} e^{\left(\frac{-12200}{T} \right)} \left[\frac{9T}{10^4} - \frac{1}{2} \right] \sqrt{[C_{12}H_{23}][O_2]} \quad (6.26).$$

This formula can only be used for temperatures above 555 (K). At lower temperatures, the fourth term on the right-hand side becomes negative and $C_{12}H_{23}$ would be formed again. However, this is not possible because reaction (6.25) is assumed to be a one-way reaction. The second category of hydrocarbon fuels consists of natural gas and user specified compositions. Here, the flow of hydrocarbons entering the combustion chamber is converted to a concentration assuming that the molar mass of the hydrocarbons is the same as the methane molar mass. The burning rate of methane is taken from (Dryer, Ref. 10):

$$\frac{d[CH_4]}{dt} = -10^{10.2} e^{\left(\frac{-48400}{RT} \right)} [CH_4]^{0.7} [O_2]^{0.8} \quad (6.27).$$

The UHC level is found by integrating equation (6.26) or (6.27). In every reactor where fuel enters, $C_{12}H_{23}$ or CH_4 is assumed to be formed.

Smoke (soot) production

A number of smoke properties are described in (Appleton, Ref. 1). It appears that the soot formed in flames only weakly depends on the conditions where it is formed. For example, the soot formation is little affected by the type of flame (premixed or diffusion). Soot primarily contains carbon, although also hydrogen and oxygen can be present. Concerning structure, soot particles are roughly spherical and grouped together in a necklace-like fashion.

The smoke emission model is based on the above mentioned statements. It combines a smoke formation model with a model describing the subsequent oxidation. The smoke formation model is an empirical equation, taken from (Rizk, Ref. 37). However, the equation is modified. The original equation is:

$$S = 0.0145 \frac{FAR_{pz} p_3^2}{F \cdot W_3 T_{pz}} (18 - H)^{1.5} \cdot \left(1 - 0.00515 \frac{e^{(0.001 T_{sz})}}{FAR_{sz}} \right) \quad (6.28),$$

where:

- FAR_{pz}, FAR_{sz} = (primary / secondary zone) fuel-air-ratio (-),
- p_3 = combustion pressure (kPa),
- F = fraction of total air used in primary zone combustion (-),
- W_3 = oxidant (often air) mass flow (kg/s),
- T_{pz}, T_{sz} = (primary/secondary) zone temperature (K),
- H = hydrogen mass percentage in fuel (-).

This equation is based on measurements in diffusion flame combustion chambers and describes soot formation as well as oxidation. Because soot formation is a relatively poorly understood process, certainly compared to soot oxidation, this formula is modified in a way only to predict soot formation (i.e. by leaving the most right term between brackets out of the equation). For soot oxidation, more extensive, less empirical equations are used. Two other changes are made to equation (6.28).

The first change made is that the fuel-air-ratio term is replaced by the equivalence ratio, multiplied with the stoichiometric fuel-air-ratio. This change serves two purposes. The first is that the formula is now also valid if the oxidant entering the combustion chamber is not air. In this case, the fuel-air-ratio can't be determined by dividing the fuel and oxidant mass flows. Now, the equivalence ratio is calculated, which is also defined if the oxidant is not air, and the stoichiometric fuel-air-ratio only depends on the fuel composition. The second purpose of this change is that the formula produces more realistic smoke formation values if low-calorific-value fuels are used. These fuels normally have a high (mass based) fuel-air-ratio, which would result

in high emission levels predicted by (6.28), because the soot produced varies linearly with the fuel-air-ratio. In reality, it is not likely that low-calorific-value fuels would produce so much more soot. In case of low-calorific-value fuels, the stoichiometric fuel-air-ratio applied is an average value for high-calorific-value fuels. In this way, the formula predicts about the same emission levels (for equal hydrogen mass percentages in the fuel) for both types of fuel. Further investigations could be used to verify if both types of fuel indeed produce equal amounts of soot.

The last change made to (6.28) is that for F, the total oxidant inserted so far into the combustion chamber is used. In this way the formula can also provide estimates of smoke formation, in combustion chambers where fuel is inserted into other zones than the primary zone. Applying these changes, formula (6.28) becomes:

$$S = 0.0145 \frac{\phi \cdot FAR_{stoich} P_3^2}{FW_3 T} (18 - H)^{1.5} \quad (6.29).$$

The second part of the smoke model describes soot oxidation. Nagle and Strickland-Constable (Ref. 33) originally developed the equations, but their validity was reinforced in (Appleton, Ref. 1). The theory is primarily valid for small soot particles. For deeper backgrounds on the model, readers are referred to (Appleton, Ref. 1).

The smoke oxidation rate is presented in the form of the overall specific surface reaction rate, which is given by:

$$\omega = 12x \left[\frac{k_A p_{O_2}}{1 + k_Z p_{O_2}} \right] + k_B p_{O_2} (1 - x) \quad (6.30),$$

where: ω = Specific surface oxidation rate (g/cm²/s);
 p_{O_2} = partial pressure of O₂ (atm);

The unknown terms in the equation can be found by applying the following equations:

$$x = \frac{1}{1 + \frac{k_T}{p_{O_2} k_B}} \quad (6.31),$$

$$k_A = 20e^{\left(\frac{-30}{RT}\right)} \quad (6.32),$$

$$k_B = 4.46 \cdot 10^{-3} e^{\left(\frac{15.2}{RT}\right)} \quad (6.33),$$

$$k_T = 1.51 \cdot 10^5 e^{\left(-\frac{97}{RT}\right)} \quad (6.34),$$

$$k_Z = 21.3 e^{\left(\frac{4.1}{RT}\right)} \quad (6.35).$$

The calculation procedure is as follows. First, the smoke formation is found by application of formula (6.29). This formula results in a smoke (mass) concentration. This concentration is converted into a number of spherical smoke particles per unit of combustion gases. This number is, of course, dependent on the radius of these spheres. The user can specify this radius. From literature, a standard value of 40 (nm) is taken.

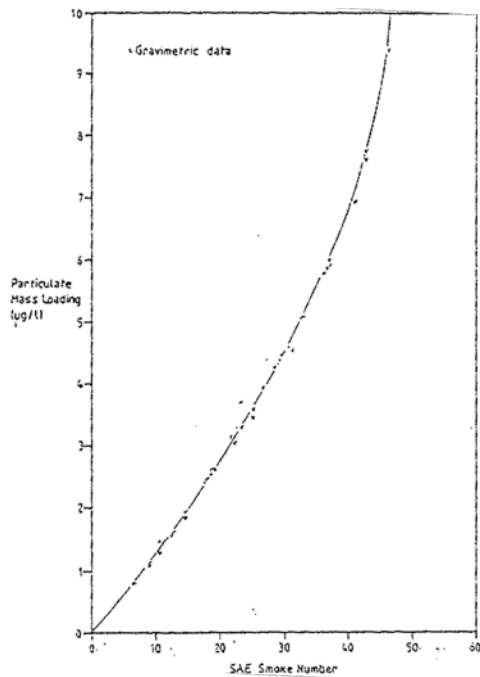


Figure 6.5 Relation between particulate mass loading and smoke number (Girling, Ref. 13)

Once the smoke particles are formed, they can be oxidised in subsequent reactors. The smoke surface oxidation rate is calculated using (6.30). Applying a constant average soot density of 1800 (kg/m³), this surface oxidation rate is converted into a rate of radius change. At the end of the combustion chamber the number of spherical particles and their radii are used to find the so-called 'particulate mass loading'. Using figure 6.5, the particulate mass loading can be converted into the smoke number.

If the fuel is not entirely injected into the primary zone, new particles can be generated in subsequent reactors, where particles formed in the primary zone have already become smaller.

In that case, particles with different radii would exist. For simplicity, the different radii are weighted averaged over all the particles, although this does introduce (limited) errors.

6.2.2.4 Tuning the emission model

The exhaust gas emission model presented so far can be used to predict values for emissions without prior knowledge of emission levels determined (e.g. measurements for certification purposes) for the gas turbine. However, GSP is primarily used for off-design gas turbine performance calculations. Therefore, emission data are known in the design point. These data can be used to improve emission predictions, by ‘tuning’ the model; i.e. factors in the model can be changed until the model reproduces the design point emissions. These factors are then kept constant (at the same value) for off-design situations.

Theoretically, the optimal situation would exist if a model clearly (and explicitly) discerns the different factors affecting emissions, like:

- fuel properties, including composition, temperature, pressure and fuel flow,
- oxidant properties, including composition, temperature, pressure and mass flow,
- geometric combustion chamber data.

In that case, the tuning can be used to find the value of factors that are constant for different working points, including application of different off-design fuels. From the above mentioned influence factors, geometric combustor data will often be constants for different working points (not in case of variable geometry combustion chambers). The influence of all the factors that differ from one working point to another should be accounted for in the emission model.

However, in the emission model presented above, not all the relevant factors are accounted for explicitly and therefore factors constant over all gas working points can’t clearly be discerned. Because of this, the tuning of the emission model is not optimal, and tuning until the design point data are reproduced will not guarantee a model that produces accurate predictions under all varying working conditions. But because the model does take a lot of factors affecting emissions into account (like fuel properties and oxidant properties), it is assumed that the model will behave well for varying working conditions.

Assuming that the geometrical data of the combustion chamber, as well as the cooling flow pattern are known, all emissions can be tuned (at the same time) by varying the length of the zones, because the zones can be chosen to suit the needs. When the zones vary, so does the division of the oxidant over the different zones. If the number of reactors is increased, the step-size in the integration will be decreased and the accuracy will be improved.

There are also a number of factors that can be used to tune the emissions separately. For each of the emissions, a so-called temperature factor can be varied. This temperature factor, that can have values between zero and one, determines the relation between the temperature used in the emission formation reactions, the calculated equilibrium temperature and the temperature of the cooling flow outside of the liner. If the temperature factor is one, the emission formation temperature is equal to the equilibrium temperature, and if it is zero, it is equal to the cooling flow temperature. The temperature varies linearly with the temperature factor. The temperature deviation from the equilibrium temperature can (for example) be due to non-equilibrium effects, non-adiabatic combustion and the cool wall layer. Especially for CO and UHC oxidation, this wall layer is important. By lowering the temperature factor and thereby the emission formation temperature, the cold layer is made more important.

For the determination of the smoke number, another tuning factor is present: the initial radius of soot particles. The standard value is set to 40 (nm). In case of bigger particles, the number of particles will be lower. If all other combustion conditions remain the same, the same radius decrease rate will be predicted, but because the number and size of the particles will have changed, the smoke number will also be different from the one found with the other initial soot particle radius.

7 Validation

In this chapter, a demonstration is given of performance and emission predictions using the new gas and combustor model. To start with, a model of the GE CF6-80C2 aircraft gas turbine is used to validate the emission model and predict effects of changing operating conditions. After that, a model of the GE LM2500-PE industrial gas turbine is used to calculate the effects of applying a low-calorific-value fuel in an industrial gas turbine designed for natural gas on performance as well as on emissions. Some remarks on the effects of low-calorific-value fuels on gas turbine performance were already made in paragraph 2.2. In GSP 7.0 only rough estimates of these effects could be made. These were achieved by lowering the combustion efficiency, until the heat release was the same as for a low-calorific-value fuel. However, in prediction of gas properties errors were made. In GSP 8.0, low-calorific-value fuels can be specified in detail and consequent effects on performance are calculated accurately.

7.1 The CF6-80C2 model

7.1.1 General

The GE CF6-80C2 is presently used, amongst others, on the Boeing B747-300/400, the Airbus A310 and the McDonnell Douglas MD-11. In figure 7.1, the general arrangement (GSP model) of the gas turbine is shown. This turbofan engine with a by-pass ratio of about 5 has a two-shaft layout. The low-pressure shaft connects the low-pressure turbine with the fan and booster, and the high-pressure shaft connects the high-pressure turbine with the high-pressure compressor. The engine pressure ratio is about 30.

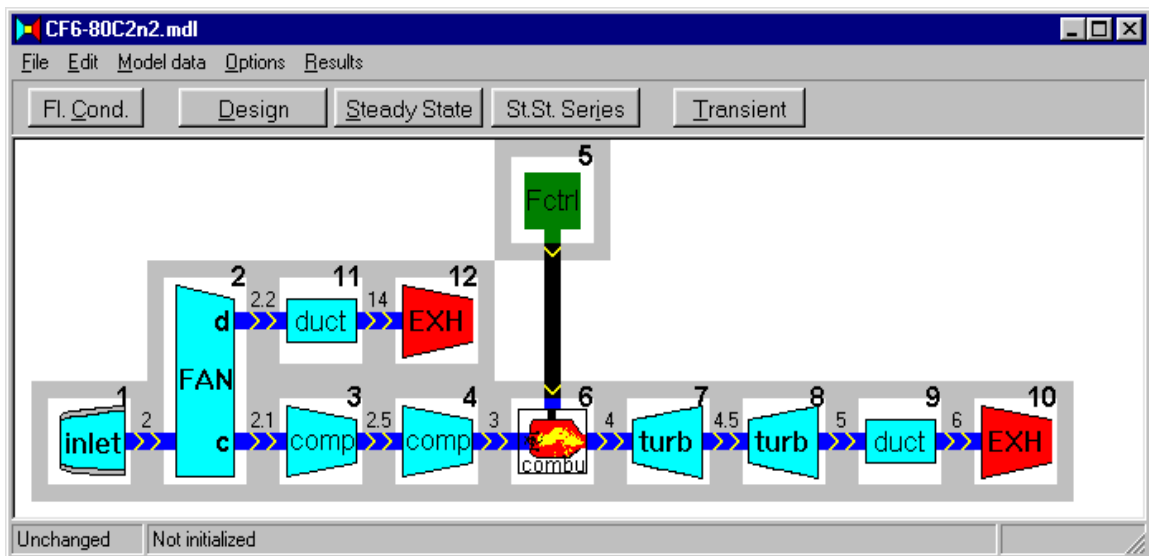


Figure 7.1 General arrangement (GSP-model) of the CF6-80C2

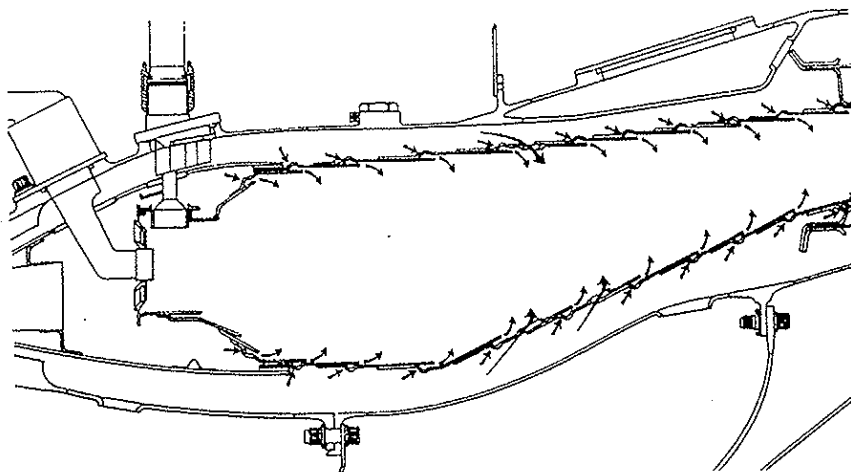


Figure 7.2 The CF-6 combustion chamber

7.1.2 Emission predictions

In order to make emission predictions, some combustion chamber data are needed. A picture of the CF-6 combustion chamber is provided in figure 7.2. It is assumed that the narrow part of the combustor, completely on the left, is used for fuel evaporation and mixing with air: because of the relatively small flow area, the speed is probably too high for stable combustion. After this narrow part, the flow area increases, enabling combustion. This increase in cross-sectional area also provides a recirculation zone. For some distance, the area remains constant achieving relatively long residence times, favourable for a good burn out. After that, the flow area decreases gradually while cooling air is added. Here, the fuel is further oxidised and the gases are accelerated to speeds favourable for the turbine.

Table 7.1 Zone data for the CF6-80C2 combustion chamber

Zone	Flow (exit) area (m ²)	Length (m)	Cooling fraction (-)
Flame front	0.360	0.0000	0.27
Primary	0.360	0.0250	0.06
Secondary	0.360	0.0743	0.22
Dilution	0.1653	0.4000	0.45

For this study, the combustion chamber was divided into the flame front and three subsequent zones. The flame (front) is assumed to be present in the beginning of the combustion chamber, where the maximum flow area is reached. The primary zone is assumed to be short and a small flow of air is added to the combustion gases. After this zone, the somewhat larger secondary zone comes, where more air is added. The point where the flow area starts to decrease, is the starting point for the dilution zone. The dimensions of the combustor were estimated using figure 7.2 and a scale picture. In table 7.1, the zone dimensions are given as well as the cooling flow division over the zones.

Table 7.2 ICAO engine exhaust emission data for the CF6-80C2 (Ref. 20)

Mode	Power Setting (%F ₀₀)	Time (min.)	Fuel flow (kg/s)	Emission indices (g/kg)			Smoke Number (-)
				UHC	CO	NO _x	
Take-off	100	0.7	2.353	0.08	0.52	28.06	7.1
Climb out	85	2.2	1.913	0.09	0.52	21.34	-
Approach	30	4.0	0.632	0.20	2.19	8.97	-
Idle	7	26.0	0.205	9.68	43.71	3.74	-

Validation

Once these data were inserted in GSP 8.0, the model was tuned to the emission values from the ICAO engine exhaust emissions data bank, given in given in table 7.2. The temperature factors for NO_x and smoke were higher than those for CO and UHC, because the cool boundary layer plays a less important role for NO_x and smoke than for CO and UHC. The soot radius of 50 (nm) was a little higher than the value of 40 (nm) found in literature.

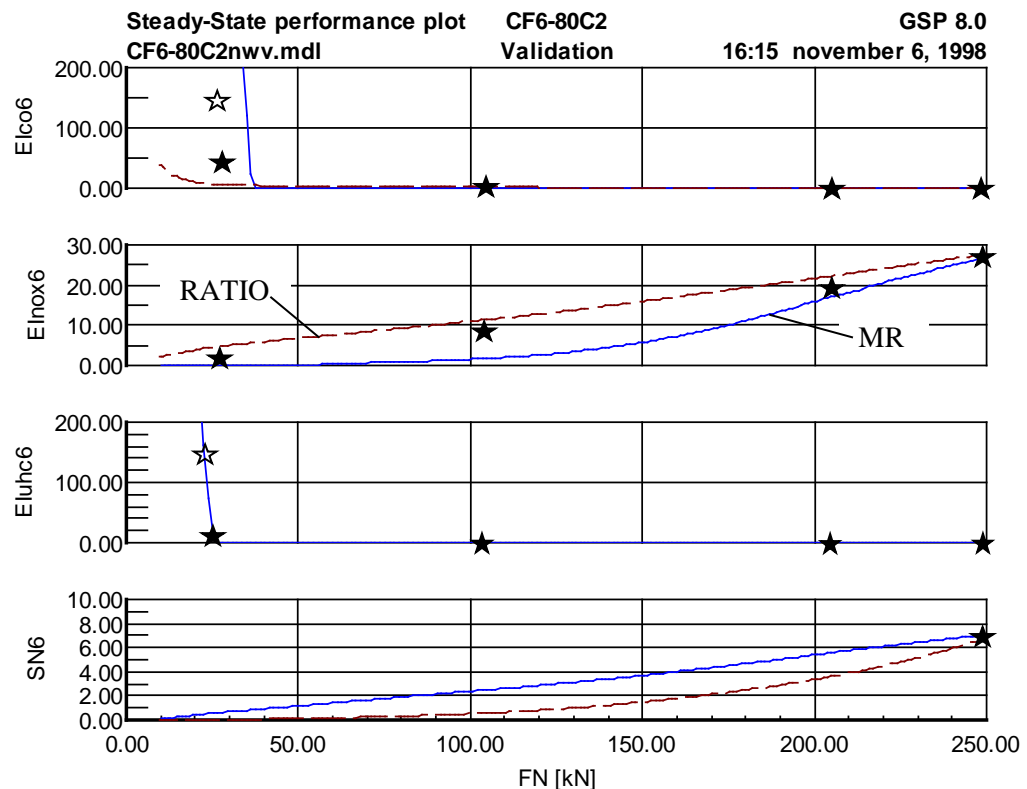


Figure 7.3 Emission predictions of the new multi-reactor model and the old ratio-model

The emission indices and smoke number predicted by the GSP 8.0 multi-reactor combustor model (MR in the figure) are shown in figure 7.3 as a function of thrust. These data are compared with the predictions of the GSP 7.0 emission ratio model (RATIO in the figure). To validate the model, data from the ICAO engine exhaust emission data bank (see table 7.2), denoted by H, and emission levels measured in a test bed, denoted by I, are also shown. It is obvious that both the ratio model and the multi-reactor model have problems to predict CO emissions at lower power settings. The multi-reactor model does predict the peak, but at a power setting higher than where the peak was measured. The NO_x -measurements are reasonably approximated for high power settings by both models. At low power settings, the ratio model appears to provide better NO_x -predictions. For unburned hydrocarbons, no ratio model is present. The multi-reactor model can well be tuned to the measured emission levels. For smoke,

only one measurement value is available: the smoke number at take-off conditions. This can be reasonably well approximated by both the models, but whether the values in other working points are well estimated, can't be seen. Further investigations will be needed to verify this. For the case considered here (a revised aircraft gas turbine burning jet fuel), both the models predict the measured emissions reasonably well. The main advantage of the multi-reactor model, however, is that the applicability is far wider than the ratio model's applicability. This is shown below by determining the effects of turbine deterioration on emissions. After that, the multi-reactor model is used to predict the effects of differing ambient conditions and application of alternative fuels on exhaust gas emissions.

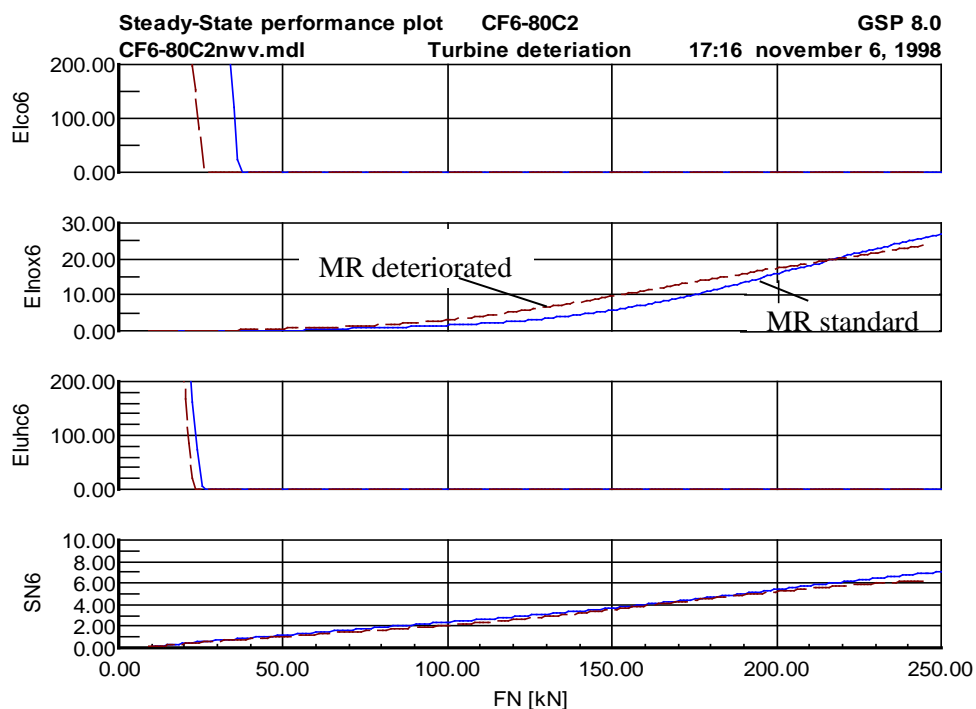


Figure 7.4 Effect of turbine deterioration on exhaust gas emissions

Turbine deterioration

In this study, the (high-pressure turbine) deterioration is modelled by an increase of 2% in turbine mass flow and a decrease of 4% in the (isentropic) turbine efficiency. In the (non-deteriorated) design point, the flame front is assumed stoichiometric.

In figures 7.4 and 7.5, the effects on emissions are shown. In both figures, the solid lines show the (GSP 8.0) multi-reactor predictions for a non-deteriorated turbine, and the dashed lines for a deteriorated turbine. In the top graph of figure 7.5, the multi-reactor results are compared with (GSP 7.0) ratio model NO_x predictions for a non-deteriorated turbine (dashed single dotted line) and a deteriorated turbine (dashed double dotted line). The ratio model predicts lower NO_x -emissions for the deteriorated engine for all power settings, which is not realistic. This shows

the limitations of ratio and related P3T3 models. These models assume fixed standard engine and operating condition relations between combustor condition parameters (i.e. P_3 , T_3 , combustor inlet mass flow and fuel flow). With deterioration these relations are changed, so more detailed models (such as multi-reactor models) are required.

In the second and third graph of figure 7.5, the (flame front) equivalence ratio and (static) temperature are shown, determined by the multi-reactor model. The multi-reactor model predicts that deterioration makes the flame front rich for high power settings, and brings the flame front closer to stoichiometric for lower power settings. This is also to be expected, because more fuel is injected to achieve equal thrust. This results in higher (flame front) temperatures for lower power settings and about the same temperatures for high power settings (see third graph). The multi-reactor model appears to predict these effects correctly.

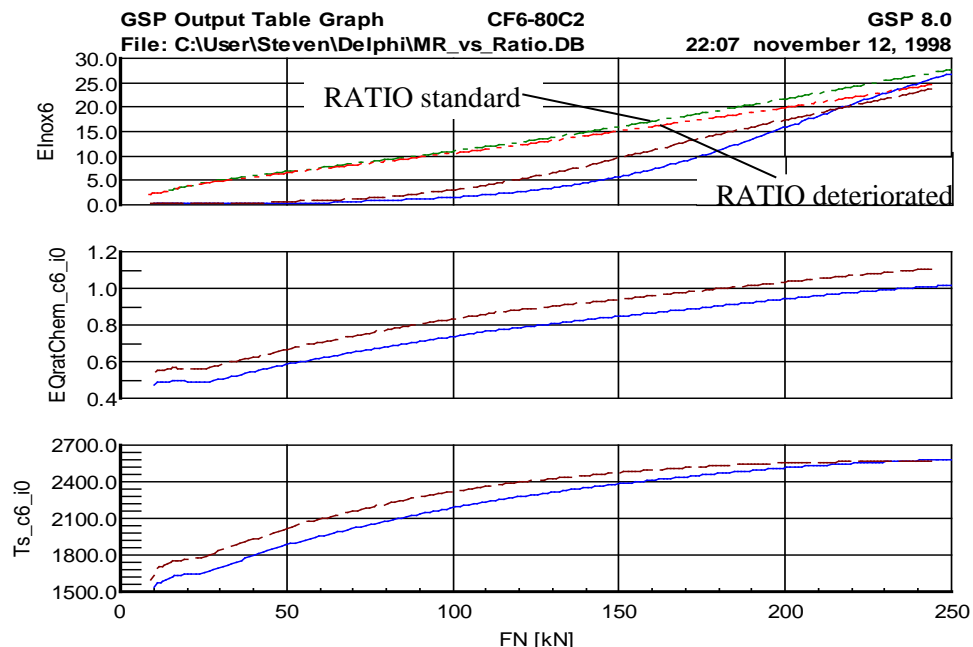


Figure 7.5 Comparison between multi-reactor model and ratio model for deteriorated turbine

The results (see figure 7.4) are a (thermal) NO_x decrease at higher power settings because of lower oxygen concentrations, and a NO_x increase for lower power settings. Also, (slightly) lower smoke emissions and a movement of the peak in CO and UHC to lower power settings are predicted.

Differing ambient conditions

In figure 7.6, the effect of ambient temperature and relative humidity on emissions is shown (as a function of thrust). The solid line gives the predictions for standard (dry air) ISA conditions. The dashed line gives the predicted emissions for a relative ambient air humidity of 100%

(denoted by Rel. H. in the figure), and the dashed dotted line gives emission predictions for an ambient temperature 10 (K) higher than ISA (denoted by ISA+10). It is obvious that the nitric oxide emissions are substantially decreased if the relative humidity increases. This was to be expected, because more water leads to lower combustion temperatures, and thus lower thermal NO_x production. This can also be the explanation for the slightly increased smoke predictions: the soot burnout is worse because of the lower temperatures. The effect on CO and UHC is limited.

The higher ambient temperatures obviously result in increased NO_x emissions. This is logical because of the higher temperatures in the gas turbine, and thus also in the combustion chamber. As expected, the higher temperatures result in lower smoke emissions, and the peak in CO and UHC production occurs at slightly lower power settings.

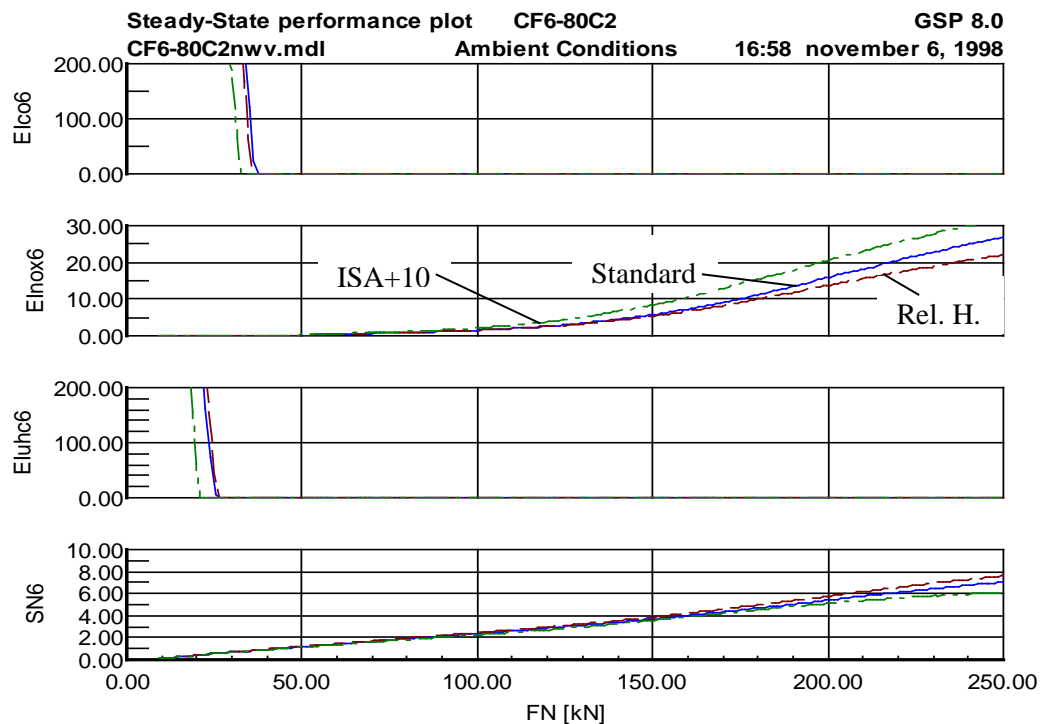


Figure 7.6 Effects of changing ambient conditions on exhaust gas emissions

Alternative fuels

Finally, the effect of applying alternative fuels on NO_x emissions was studied. The results are shown in figure 7.7. The solid lines are emission predictions for Jet-A, the dashed lines for methane and the dashed dotted lines for hydrogen. Both methane and hydrogen were supplied at a temperature of 288.15 (K) (the same supply temperature as Jet-A) in gaseous form.

For low power settings, hydrogen gives the lowest amount of NO_x emissions, and for high power settings the highest amount of NO_x emissions. The NO_x emissions achieved by burning methane are for all power settings lower than those achieved when burning Jet-A. These results were further studied by looking at the (static) temperature in the flame front (not necessarily stoichiometric) and the equivalence ratio in the flame. Although the other zones are also important for NO_x -formation, here only the flame temperature and equivalence ratio are shown, because in the flame the highest NO_x -formation speeds are encountered. The difference between the emission index for methane and Jet-A is due to the lower flame temperature; the equivalence ratios are practically the same. For hydrogen however, the flame temperature is lower than the flame temperature for Jet-A, but so is the equivalence ratio. Therefore, more oxygen is available in the hydrogen flame, resulting in higher thermal NO_x -formation.

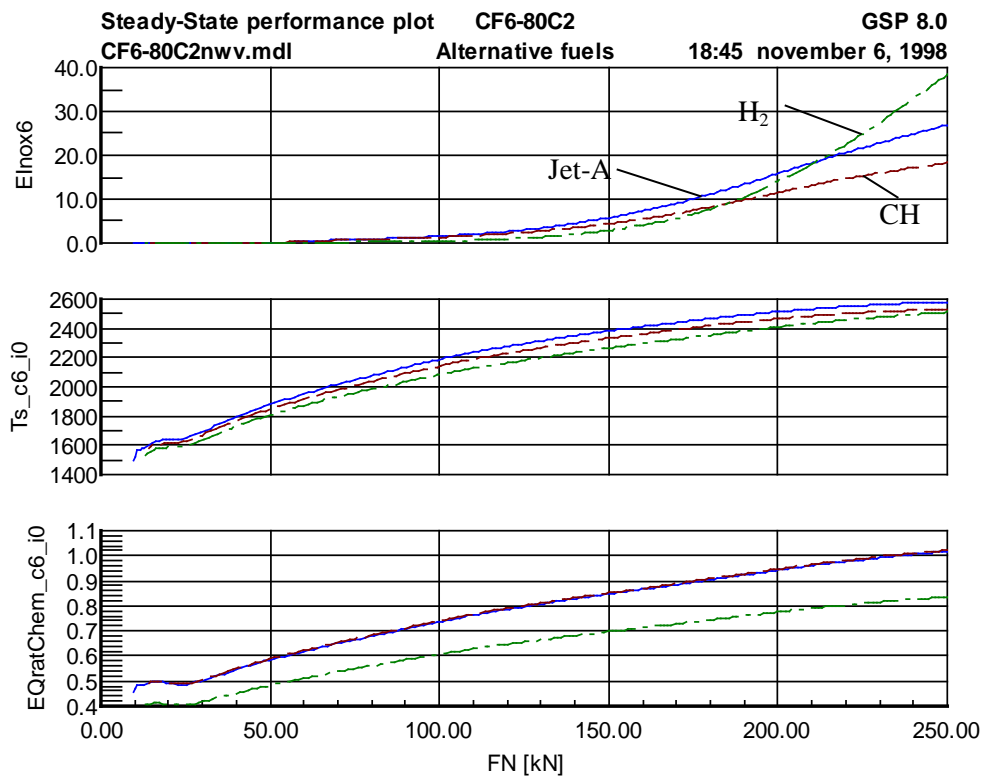


Figure 7.7 Effects of applying alternative fuels (methane and hydrogen) on NO_x

From this investigation, it could be concluded, that hydrogen combustion results in higher NO_x emissions for high power settings. However, this is not (entirely) true. The reason for this is that a few things are neglected in this study. The first thing is the fact that hydrogen has a higher burning speed than Jet-A. For this reason, preferably a different (smaller) combustion chamber should be designed, with smaller flow areas, resulting in higher flow speeds, lower residence times and thus lower NO_x values. The second thing is that, also because of a different combustion chamber applied, the flow pattern could very well be different resulting in a higher equivalence ratio in the flame zone. Because this results in a higher temperature, but also a lower amount of oxygen, it is not clear if this will mean an increase or decrease in NO_x (around stoichiometric equivalence ratios an increase will be more likely). A third thing is that hydrogen can also be fed in liquid form. In that case, a heat exchanger can be used to evaporate the hydrogen. This heat exchanger draws the energy necessary for evaporation from the air coming from the compressor, resulting in lower combustion temperatures, and consequently lower NO_x emissions.

7.2 The GE LM2500 model

7.2.1 General

The GE LM2500 is a single-shaft industrial gas turbine, derived from the CF6 family of aircraft gas turbines. There are several types of this gas turbine, delivering different shaft powers. Here, the GE LM2500-PE is studied. The mass flow handled by this engine is 68 (kg/s) and the design engine pressure ratio is 18.65. The layout of the gas turbine is shown in figure 7.8. The second turbine is a free-power turbine, which can be connected to (e.g.) an electricity generator.

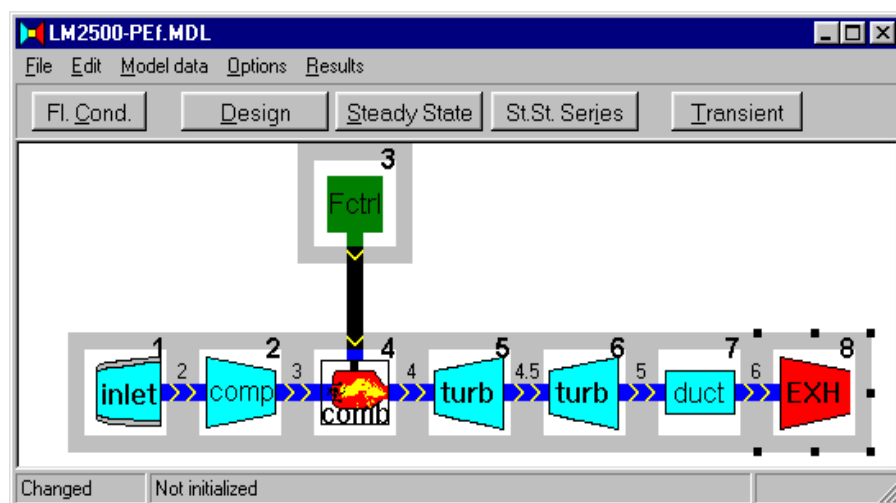


Figure 7.8 GSP model of the GE LM2500

7.2.2 Performance predictions

The gas turbine is designed for natural gas. According to GE, the molar mass of the fuel was 16.04 (g/mole) and the heating value 47680 (kJ/kg). This molar mass is the value for methane, but the heating value is somewhat lower than the heating value of methane. This indicates that the fuel probably contains some larger hydrocarbons and probably inert components like nitrogen.

Table 7.3 Composition of biomass gasification products

Component	Volume fraction (%)
CO ₂	15.6
CO	8.8
H ₂ O(g)	24.0
H ₂	7.4
CH ₄	5.2
C ₂ H ₆	0.3
C ₂ H ₄	1.1
N ₂ (incl. Ar)	37.6

The off-design fuel consists of biomass gasification products, obtained from the biomass gasifier at the Laboratory for Thermal Power Engineering at Delft University of Technology (see Hoppesteijn, Ref. 19; De Jong, Ref. 22). In table 7.3, the composition of the gasification products is given. The gases contain a lot of water and nitrogen gas, and as a result the fuel heating value is rather low: about 4 (MJ/kg) at 288.15 (K).

When low-calorific-value fuels are used in gas turbines, the fuel flows are relatively high, resulting in a large compression power needed. In this study, it was assumed that the air fed to the gasifier was compressed to 5 (bar) with an isentropic efficiency of 0.9. At this pressure, the gasification process was assumed to take place. Subsequently, the fuel was further compressed from 5 (bar) to the pressure needed for injection into the gas turbine.

The compression power needed to compress the low-calorific-value fuel to the injection pressure is calculated within GSP itself. Assumed was an isentropic compression efficiency of 0.85. The compression power needed to compress the air from 1 to 5 (bar) was partly hand calculated using formulae (C.44) and the following expression:

$$P_{comp} = m_{air} \cdot c_p \cdot \Delta T_{is} / \eta_{comp} \quad (7.1)$$

Because the amount of air needed (m_{air}) is unknown in the program, an attempt was made to relate it to the fuel flow. This was done by assuming that all the nitrogen in the low-calorific-value fuel (see table 7.3) originates from the air used in the gasification process. During the gasification process, the air is assumed to be diluted by other gasses, generated by the gasification. In that case, the relation between the amount of air needed and the fuel flow is $37.6/78.084 = 0.48$ (kg/s) of air needed per (kg/s) fuel (the 78.084 is the volume percentage of nitrogen in air). For $\gamma = 1.4$, the compression process involves an isentropic temperature change from 288.15 (K) to 456.38 (K) (using equation (C.44)). Together with assumed values of 0.9 for the compressor efficiency and 1005 for the specific heat at constant pressure, the compression power needed (in MW) equals 0.090 times the fuel flow. Both compression powers described were subtracted from the power delivered by the power turbine. In figure 7.9, the GSP results are given.

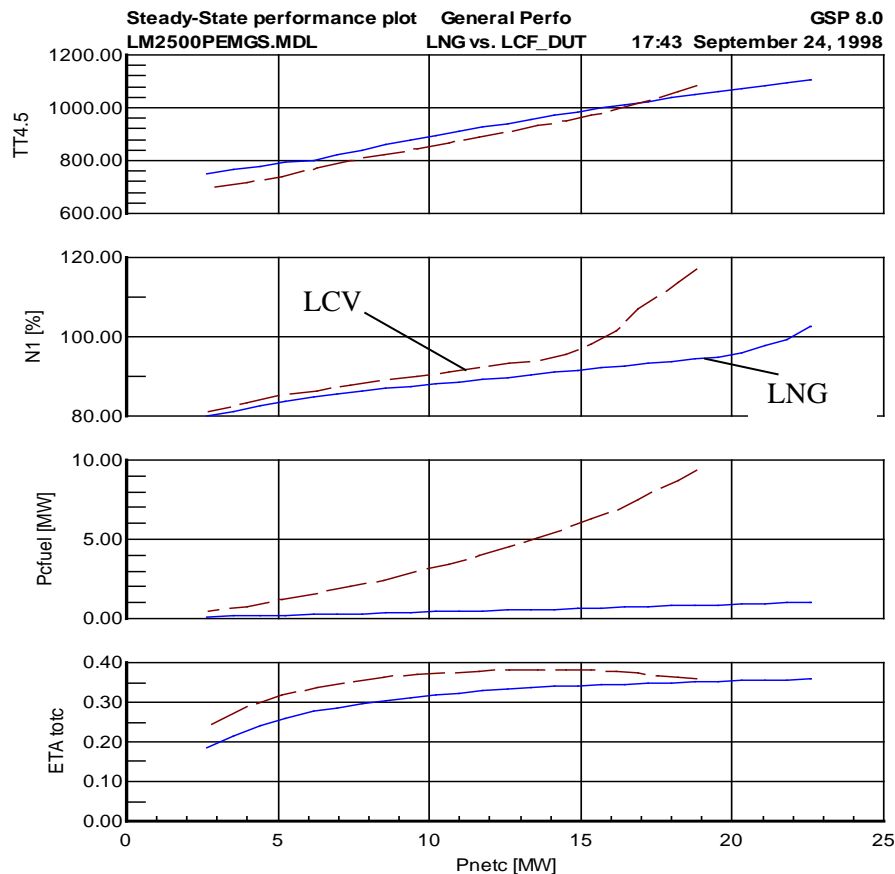


Figure 7.9 Effects of low-calorific-value alternative fuel on performance

In this figure, the solid lines show the results with natural gas (LNG) and the dashed lines with the low-calorific-value (LCV) biomass fuel. Pnetc is the power that is left, when both aforementioned compression powers are subtracted from the power generated by the power

turbine. The total inlet temperature of the power turbine (TT4.5) is about the same for both fuels. However, the second figure shows, that the rotor speed of the power turbine becomes too high, if the same power is to be generated using low-calorific-value fuel. This is due to the large mismatch between the compressor and the high-pressure turbine, resulting from the large fuel flow injected. In order to obtain the higher power output, the gas turbine hardware could be modified (e.g. by an increase in turbine flow capacity) or the compressor load could be increased (e.g. by taking compressor bleed air to feed the gasifier). The power needed to compress the fuel from 5 bar to the pressure in the combustion chamber (P_{cfuel} , see third graph) is far higher for low-calorific-value fuel than for natural gas. However, this does not result in lower cycle efficiency (η_{Atotc} , see fourth graph), if the power delivered is not to become too high, although the cycle efficiency may well have to be corrected with extra power required for the gasifier.

In figure 7.10, the effect of applying the alternative fuel on compressor performance is given in the compressor map. The working line with biomass fuel is the line closest to the stall line. Although the difference with the working line for natural gas is not very big, hardware modifications may be necessary to retain the proper stall margin.

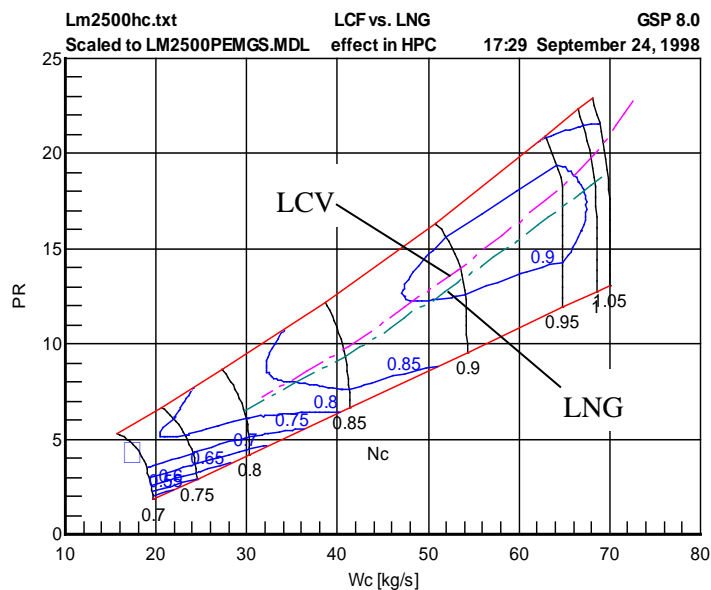


Figure 7.10 The effect of applying low-calorific-value fuels on compressor operation

To avoid the high rotor speeds, a fixed power turbine (i.e. single shaft engine) could be used. However, further studies (not shown here) showed that a fixed power turbine leads to severe stall problems. For a more complete picture of gas turbine performance using low-calorific-value fuels, readers are referred to (Locadia, Ref. 29).

7.2.3 Emission predictions

Secondly, the LM2500-PE industrial gas turbine model described above was used to investigate the effects of applying alternative fuels in (industrial) gas turbines on the NO_x and CO emissions. The multi-reactor emission model used had similar characteristics as those of the CF6-80C2 model, assuming similar combustor geometry. The results are shown in figure 7.11.

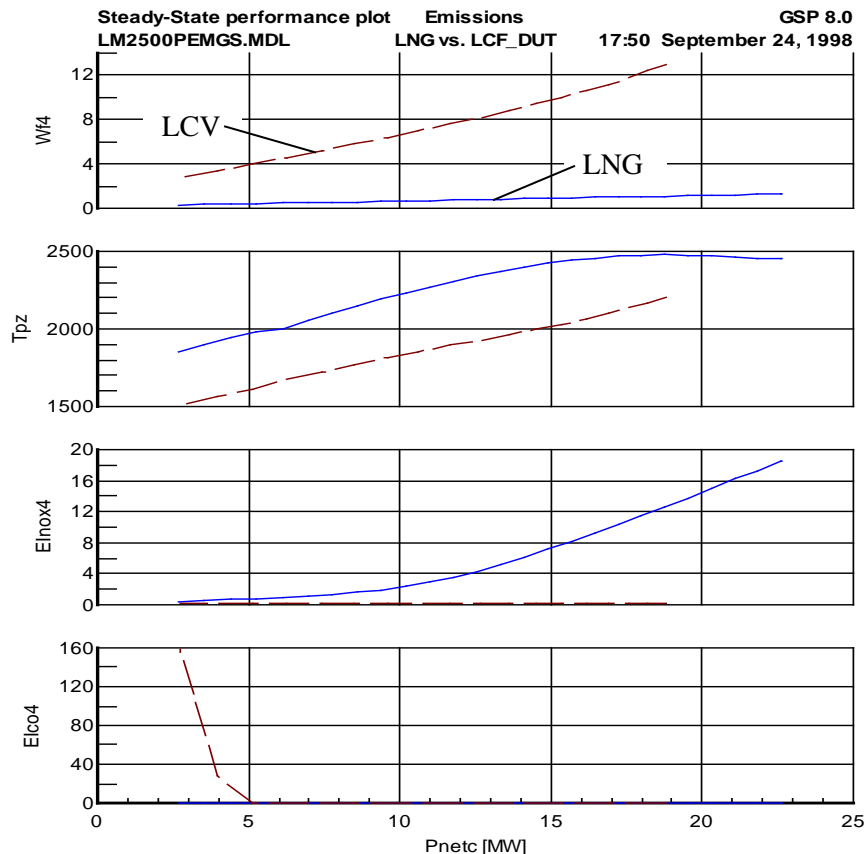


Figure 7.11 Emission data for the LM2500-PE industrial gas turbine

The relevant properties are given as a function of P_{netc} , the power of the free power turbine minus the power needed to compress the fuel, like in paragraph 7.2.2. First, the fuel flow necessary to generate the power is given. To find the total pollutant emissions, the emission indices must be multiplied with this number. In the second diagram, the primary zone temperature is shown. It is obvious that combustion of low-calorific-value fuel involves a lower primary zone temperature. This is an explanation for the lower NO_x emissions predicted for low-calorific-value fuel, shown in the third diagram. However, the low-calorific-value fuel used here did not contain fuel-bound-nitrogen: it was assumed that the nitrogen containing molecules would be removed before combustion. If this would not be the case, higher NO_x emissions would result. GSP can be used to predict emission levels, when more fuel-bound-nitrogen is

present, but this was not done here. In the fourth diagram, it can be seen that relevant values for CO emissions are only predicted for low-calorific-value fuels. This is probably caused by the lower (adiabatic flame) temperatures in the combustion chamber, inhibiting CO to oxidise completely.

7.3 Discussion on the results

The emission predictions for the CF6-80C2 were (reasonably) in accordance with the measured data after tuning. In the subsequent investigation of effects of ambient conditions, engine deterioration and alternative fuels, expected effects on emissions were found. Therefore, it has become clear, that GSP has become a powerful tool to predict the effects of operating conditions on exhaust gas emissions. In the investigations using the GE LM2500-PE model, it has become clear that the applicability of GSP has been largely extended. The effects of applying different off-design fuels on gas turbine performance can now be studied in a reliable way.

During the validation, however, it has become clear that the combustor model has still got some (important) shortcomings. In general, when tuning the combustor model, it was observed that all the emissions were extremely dependent on combustor geometry, choice of zones, cooling flow distribution and other tuning factors. A probable explanation for this is that the number of zones has been chosen too small, resulting in large integration steps. These large integration steps result in low accuracy and great sensitivity to boundary conditions. The linearisation applied is especially inaccurate if the equations integrated are highly non-linear, which could very well be the case for the emission indices and smoke number.

It has become clear that the NO_x -emission modelling produces realistic results when off-design working points are calculated. Only for the lower power settings, the amount of nitric oxides is underestimated. This could be due to limited validity of the reaction kinetic data used. Another probable explanation is that the cooling flow division over the zones changes at lower power settings, keeping the first zones around stoichiometric equivalence ratios, resulting in relatively high NO_x -emissions. The combustor model predicts equivalence ratios substantially lower than one for low power settings. A better combustor flow model could be used to verify this. Because no very rich zones were investigated, the validity of the square root dependence of prompt NO_x on pressure could not be verified. Further research will be needed for this.

The carbon monoxide emission prediction can be used qualitatively, but needs some changes until it can be used quantitatively. Now, no carbon monoxides are predicted at high power settings, while at low power settings (very) high carbon monoxide emissions are predicted. In reality, there is no sudden peak, but first a gradual built up of CO emissions as the power setting

is lowered. The most probable cause for this is that the integration steps are too big: once a formation speed is predicted that is too high or too low, it has a huge influence on carbon monoxide levels. Normally high depletion speeds cause a quick *temporary* depletion. Once the quantity formed decreases, the depletion speeds get lower. Here, due to the large step size, it takes some time until the formation speed is recalculated. By that time, the amount of predicted carbon monoxide is already very low. It should be noted, that in literature, carbon monoxide emissions are often predicted using a well-stirred reactor, as is also the case in GSP, followed by a series of (one dimensional) plug flow reactors, which is not the case in GSP. These plug flow reactors give a better, more detailed description of carbon monoxide oxidation. In GSP, a good option for better CO oxidation modelling is taking small integration steps in the lower temperature (dilution) zones. In the higher temperature zones, the equilibrium amounts of CO can be used. These serve as the start value for a detailed CO integration in lower temperature zones.

Results not shown here indicate that the prediction of unburned hydrocarbon emission indices gives some problems for fuels other than jet fuels and diesel. If a jet fuel (or diesel) is selected, the dependence of the unburned hydrocarbons on power setting and the temperature factor is comparable to the carbon monoxide case. However, for other fuels, the temperature factor must be high in order to predict low emission levels at high power settings. Therefore, a better methane oxidation description may be necessary. If the amount of unburned hydrocarbons is to be better approximated, the integration step size should be chosen smaller, but even better would be to use a more extended kinetic scheme.

Finally, the smoke number appears to be reasonably approximated. For low power settings, the predicted smoke number is low and it becomes higher for higher power settings. Further investigation has to show whether the smoke number predicted as a function of power setting is good.

8 Conclusions and Recommendations

The Gas turbine Simulation Program (GSP), after being extended with a gas model accurately describing compositions and calculating equilibrium compositions, is a powerful tool to predict effects of operating conditions such as alternative fuels as well as water/steam injection on gas turbine performance and emissions. This is a significant improvement compared to the last version of GSP, which (ratio) emission model had a very limited applicability in predicting effects of operational variables.

The new gas model was successfully demonstrated in the analysis of the effects of low-calorific-value gas from a biomass gasifier on various performance parameters and emissions. This type of performance analysis can be used to support decisions concerning engine hardware modifications.

The new multi-reactor combustor model is a generic structure in which (kinetic) models describing formation of species, including exhaust gas emission species, can easily be implemented.

For NO_x , CO, UHC and smoke, models have been developed for instantaneous formation in the flame zone and subsequent formation (/depletion) according to multi-reactor kinetic schemes. The multi-reactor model is best used as sensitivity analysis tool; i.e. to calculate effects on performance and emission parameters relative to reference values. The emission models have been demonstrated on a large turbofan engine. The results corresponded well with measured emission data and with the expected operating condition effects on emissions.

Improvements in the emission models can be made by better modelling of effects not covered in a one-dimensional model such as film cooling and division of cooling flows over the combustion zones. The accuracy of the predictions could probably be improved by adapting a smaller step size for the numerical integration. To increase the applicability of the combustor model, the procedure used to calculate the equilibrium composition at given temperature could be changed. Low O_2 -concentrations would not pose problems anymore then. In general, more work needs to be done to validate results, preferably using detailed combustor (flow) data of a variety of engines and operating conditions.

The chemical model can easily be extended with a limited number of species. For turbo-shaft emissions calculations the chemical model has been extended with the specie of N-radicals. However, this only led to minor changes in the emission-indices. However the inclusion of the combined effect of mixture and chemical reactions led to a substantial improvement in the calculation of the emission indices.

In order to improve the emission model, some recommendations can be made:

- in general, more testing is needed, preferably using more off-design emission levels, to verify the results obtained using the combustor model,
- a better flow model is needed to predict the off-design flow division of the cooling air over the different reactors,

- an integration procedure using small steps should be tested to find out whether this will improve the accuracy of predictions,
- a better modelling of the effects of the cool boundary layer on especially carbon monoxide and unburned hydrocarbon emissions is needed,
- possibly, if the carbon monoxide and unburned hydrocarbon emission predictions are still not satisfying, more extended kinetic schemes could be implemented for calculation of these emissions,
- the calculation of equilibrium in the combustion chamber should be changed in order to make low O_2 -concentrations possible. As an alternative the method of the minimisation of the Gibbs energy can be implemented (as in NASA CEA) to resolve combustion equilibrium. Main advantages of the NASA method are the treatment of fuel rich mixtures and the ease of treatment by an extension of the number of chemical species.

References

- 1) Appleton, J.P., 'Soot oxidation kinetics at combustion temperatures', *Atmospheric Pollution by Aircraft Engines, AGARD-Conference Proceedings-125*, paper 20. Neuilly Sur Seine: AGARD, 1973.
- 2) Bachmaier, F., Eberius, K.H., Just, Th., 'The formation of Nitric Oxide and the detection of HCN', *Combustion Science and Technology*, vol.7, p.77. New York: Gordon and Breach Science Publishers Ltd., 1973.
- 3) Barrère, M., 'Modélisation des foyers de turboréacteur en vue de l'étude de la pollution', *Atmospheric Pollution by Aircraft Engines, AGARD-Conference Proceedings-125*, paper 27. Neuilly Sur Seine: AGARD, 1973. (in French)
- 4) Bokhorst, E.C., *The use of alternative gaseous fuels in heavy-duty gas turbines. The effect on gas turbine operation in general and NO_x emissions in particular, EV 1844*. Delft: Delft University of Technology, 1995.
- 5) Botros, M.J., et al., 'One-dimensional predictive emission monitoring model for gas turbine combustors', ASME Paper 97-GT-414, *ASME Technical Papers*, New York: ASME, 1997.
- 6) Bozza, F., Tuccillo, R., Fontana, G., 'Performance and Emission Levels in Gas Turbine Plants', *Journal of Engineering for Gas Turbines and Power*, vol.116, p.53-62. New York: ASME, 1994.
- 7) Bruin, M., *Exhaust gas emission models in the NLR gas turbine simulation program GSP 7.0, memorandum VH-96-009*. Amsterdam: National Aerospace Laboratory, 1996.
- 8) Chleboun, P.V., Hubbert, K.P., Sheppard, C.G.W., 'Modelling of CO Oxidation in Dilution Jet Flows', *Combustion and Fuels in Gas Turbine Engines, AGARD-Conference Proceedings-422*, paper 38. Neuilly Sur Seine: AGARD, 1988.
- 9) Cohen, H., Rogers, G.F.C., Saravanamuttoo, H.I.H., *Gas Turbine Theory*. London: Longman Group Limited, 1996.
- 10) Dryer, F.L., Glassman, I., 'High-Temperature oxidation of CO and CH₄', *Pollutant Formation and Destruction in flames, 14th Symposium (Int.) on Combustion*, p.987-1003. Pittsburgh: The Combustion Institute, 1973.
- 11) Fenimore, C.P., 'Formation of Nitric Oxide from Fuel Nitrogen in Ethylene Flames', *Combustion and Flame*, vol.19, p.289-297. New York: American Elsevier Publishing Company, Inc, 1972.
- 12) Fletcher, R.S., Heywood, J.B., 'A model for nitric oxide emissions from aircraft gas turbine engines', AIAA Paper 71-123, *AIAA Technical Papers*. New York: AIAA, 1971.

- 13) Girling, S.P., 'Gas turbine smoke measurement: a smoke generator for the assessment of current and future techniques', *Combustion and Fuels in Gas Turbine Engines, AGARD-Conference Proceedings-422*, paper 20. Neuilly Sur Seine: AGARD, 1988.
- 14) Glassman, I., *Combustion*, Princeton: Academic Press, 1996.
- 15) Gordon, S., McBride, B.J., *Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications. I. Analysis*. NASA Reference Publication 1311. Ohio, National Aeronautics and Space Administration, Lewis Research Center, 1994.
- 16) Hammond, D.C. JR., Mellor, A.M., 'Analytical Calculations for the Performance and Pollutant Emissions of Gas Turbine Combustors', *Combustion Science and Technology*, vol.4, p.101-112. New York: Gordon and Breach Science Publishers Ltd., 1971.
- 17) Hein, K.R.G., Stencils for the course 'Fuel Conversion'. Delft: Delft University of Technology, Faculty of Mechanical Engineering and Marine Engineering, 1997.
- 18) Holderness, F.H., Macfarlane, J.J., 'Soot Formation in Rich Kerosine Flames at High Pressure', *Atmospheric Pollution by Aircraft Engines, AGARD Conference Proceedings-125*, paper 18. Neuilly Sur Seine: AGARD, 1973.
- 19) Hoppesteyn, P.D.J., Andries, J., Hein, K.R.G., 'Biomass/coal derived gas utilization in a gas turbine combustor', ASME Paper 98-GT-160, *ASME Technical Papers*. New York: ASME, 1998.
- 20) 'ICAO Engine Exhaust Emissions Data Bank', issue 1, 1993, Appendix C, data for CF6-80C2B1F, page C-41.
- 21) Jentink, H.W., Veen, J.J.F. van, 'In flight spectroscopic aircraft emission measurements'. To be published in the proceedings of the symposium *Gas Turbine Engine Combustion, Emissions and Alternative fuels*. Neuilly Sur Seine: AGARD, 1998.
- 22) Jong, W. de, Andries, J., Hein, K.R.G., 'Coal-biomass gasification in a pressurized fluidized bed gasifier', ASME Paper 98-GT-159, *ASME Technical Papers*. New York: ASME, 1998.
- 23) Kan, J. van, *Numerieke Wiskunde voor technici*. Delft: Delftse Universitaire Pers, 1996. (in Dutch)
- 24) Keck, J.C., 'Rate-controlled constrained chemical equilibrium theory of chemical reactions in complex systems', *Progress in Energy and Combustion Science*, vol.16, p.125-154. Oxford: Pergamon Press plc., 1990.
- 25) Kelsall, G.J., et al., 'Combustion of LCV Coal Derived Fuel Gas for High Temperature, Low Emissions Gas Turbines in the British Coal Topping Cycle', ASME Paper 91-GT-384, *ASME Technical Papers*. New York: ASME, 1991.
- 26) Kuo, K.K., *Principles of Combustion*. New York: John Wiley & Sons, 1986.

- 27) Lavoie, G.A., Heywood, J.B., Keck, J.C., 'Experimental and Theoretical Study of Nitric Oxide formation in Internal Combustion Engines', *Combustion Science and Technology*, vol.1, p.313-326. New York: Gordon and Breach Science Publishers Ltd., 1970.
- 28) Lefebvre, A.W., *Gas Turbine Combustion*. Washington: Hemisphere Publishing Corporation, 1983.
- 29) Locadia, U.E.L., *Evaluation with GSP of the effects on the performance of the LM2500-PE gas turbine using a low calorific alternative fuel and testing of the GSP emission model*. Delft: Delft University of Technology, 1998.
- 30) McBride, B.J., Gordon, S., *Computer Program for Calculation of Complex Chemical Equilibrium Compositions and Applications. II. Users Manual and Program Description*. NASA Reference Publication 1311. Ohio, National Aeronautics and Space Administration, Lewis Research Center, 1996.
- 31) Michaud, M.G., Westmoreland, P.R., Feitelberg, A., 'Chemical mechanisms of NO_x formation for gas turbine conditions', *Twenty-fourth Symposium (International) on Combustion*, p.879-887, Pittsburgh: The Combustion Institute, 1992.
- 32) Miller, J.A., Bowman, C.T., 'Mechanism and modelling of nitrogen chemistry in combustion', *Progress in Energy and Combustion Science*, vol.15, p.287-338. Oxford: Pergamon Press plc, 1989.
- 33) Nagle, J., Strickland-Constable, R.F., 'Oxidation of carbon between 1000°C-2000°C', *Proceedings 5th Conference on Carbon*, p.154, Pergamon, 1961.
- 34) Nakata, T., et al., 'Experimental Evaluation of a Low NO_x LBG Combustor Using Bypass Air', ASME Paper 90-GT-380, *ASME Technical Papers*. New York: ASME, 1990.
- 35) Peeters, T.W.J., Roekaerts, D.J.E.M., *Turbulent Reagerende Stromingen*. Delft: Delft University of Technology, Faculty of Technical Physics, 1997. (in Dutch)
- 36) Pratt, D.T., 'Coalescence/Dispersion Modelling of Gas Turbine Combustors', *Combustor Modelling, AGARD-Conference Proceedings-275*, paper 15. Neuilly Sur Seine: AGARD, 1980.
- 37) Rizk, N.K., Mongia, H.C., 'Emissions Predictions of Different Gas Turbine Combustors', AIAA Paper 94-118, *AIAA Technical Papers*. New York: AIAA, 1994.
- 38) Rodriguez, C.G., O'Brien, W.F., 'Unsteady, finite-rate model for application in the design of complete gas-turbine combustor configurations', *Design principles and methods for aircraft gas turbine engines, AGARD Conference Proceedings*. Neuilly Sur Seine: AGARD, to be published.
- 39) Ruijgrok, G.J.J., *Elements of Airplane Performance*. Delft: Delft University Press, 1994.
- 40) Sato, M., et al., 'Coal Gaseous Fueled, Low Fuel-NO_x Gas Turbine Combustor', ASME Paper 90-GT-381, *ASME Technical Papers*. New York: ASME, 1990.

- 41) Sturgess, G.J., McKinney, R., Morford, S., 'Modification of Combustor Stoichiometry Distribution for Reduced NO_x Emission From Aircraft Engines', ASME Paper 92-GT-108, *ASME Technical Papers*, New York: ASME, 1992.
- 42) Toof, J.L., 'A Model for the Prediction of Thermal, Prompt, and Fuel NO_x Emissions From Combustion Turbines', ASME Paper 85-GT-29, *ASME Technical Papers*. New York: ASME, 1985.
- 43) Visser, W.P.J., *Gas turbine Simulation Program*, NLR CR91022L. Amsterdam: NLR, 1990.
- 44) Westenberg, A.A., 'Kinetics of NO and CO in Lean, Premixed Hydrocarbon-Air Flames', *Combustion Science and Technology*, vol.4, p.59-64. New York: Gordon and Breach Science Publishers Ltd., 1971.
- 45) Young, J.B., 'Non-equilibrium wet steam flow in low pressure turbines' in *Aerothermodynamics of Low Pressure Steam Turbines and Condensers*. Washington: Hemisphere Publishing Corporation, 1987.

Appendix A Graduation assignment

-81-

MEMORANDUM VH-98-010



A Graduation assignment



Faculteit der Werktuigbouwkunde
en Maritieme Techniek

Vakgroep Proces en Energie
Sectie Energievoorziening
Mekeiweg 2
Postbus 5037, 2600 GA DELFT
Tel. (015) 278 6690
Fax. (015) 278 2460

OPDRACHT

Classificatie : afstudeeropdracht
Opdracht nr. : EV-452
Datum : 24-2-1998

Naam : S.C.A. Kluiters
Docent : prof. ir. J.P. van Buijtenen
Begeleider(s) : ir. W.P. Visser (NLR)

Onderwerp: *Het ontwikkelen van modellen voor de gaseigenschappen ten behoeve van het Gasturbine Simulatie Programma GSP.*

Inleiding

Door het NLR en de TUD wordt het Gasturbine Simulatie Programma (GSP7.0) gebruikt voor analyse van gasturbineprestaties. Het toepassingsgebied omvat tot nu toe vooral operationele aspecten zoals diagnostiek, failure analysis e.d. waarbij dynamisch en statisch gedrag bij variërende bedrijfscondities berekend worden. Een nieuwe toepassing van GSP is de berekening van uitlaatgasemissies bij verschillende bedrijfscondities. GSP7.0 bevat een aantal eenvoudige empirische relaties voor berekening van emissieindices. De wens bestaat de emissiemodellen te verfijnen waardoor nauwkeuriger voorspellingen van emissieindices mogelijk worden en de modellen ook bruikbaar worden voor industriële gasturbines, al dan niet voorzien van water- of stoominjectie. Hiervoor is een uitbreiding van het GSP gasmodel nodig, in die zin dat ook effecten van dissociatie bij hogere temperaturen, van de aanwezigheid van water- en brandstofdamp of -druppels in het gas en van een wisselende brandstofsamenstelling gemodelleerd kunnen worden.

Opdracht

Uw werkzaamheden bestaan, gezien het voorgaande, uit:

1. Een kennismaking met GSP in de vorm van een analyse van de effecten van het toepassen van laagcalorische brandstof in een bestaande gasturbine (voor normale brandstof).
2. Een vergelijking van het gasturbinegedrag (met name de "stall-margin") voor brandstoffen met sterk uiteenlopende verbrandingswaarden.
3. Het bepalen van de tekortkomingen van GSP voor toepassing op afwijkende brandstofsamenstellingen.
4. Een literatuuronderzoek naar gasmodellen voor gasturbineprestatieberekeningen.
5. Een evaluatie van geschiktheid van deze gasmodellen voor emissieberekeningen bij alternatieve brandstoffen en -cycli voor implementatie in GSP.
6. De keuze van het gasmodel en keuze van de wijze van implementatie (bijv. tabellen of polynomen).
7. Verantwoording van het gekozen gasmodel en de gekozen aanpak voor implementatie (afweging kosten - baten voor verschillende niveaus van nauwkeurigheid).
8. Het opstellen van algoritmen voor het nieuwe gasmodel.
9. Implementatie in GSP.
10. Uitbreiden van minimaal een van de bestaande GSP - emissiemodellen op basis van het nieuwe gasmodel.
11. Validatie van resultaten aan de hand van een aantal berekeningen van emissies van een industriële gasturbine voor verschillende brandstofsoorten.

Rapportage

De resultaten van de studie dienen te worden vastgelegd in een Engelstalig rapport¹, voorzien van een copie van deze opdracht en een summary van opzet en resultaten van het onderzoek.

Hoogleraar,

Prof. ir. J.P. van Buijtenen

Begeleider,

ir. W.P. Visser

1. Van het rapport moet een gebonden exemplaar tezamen met het losbladige origineel in de bibliotheek van de Sectie worden opgenomen. Wanneer dit uit een oogpunt van vertrouwelijkheid van de informatie niet gewenst is, dient de dan te volgen handelwijze met docent of TUD-begeleider te worden vastgesteld.
Het schutblad van het rapport - in ieder geval van de archiefexemplaren - dient volgens de voorschriften te worden ingericht. Het rapportnummer wordt verstrekt door het secretariaat.

Appendix B Combustion chemical reactions

B.1 Reaction kinetics

In a gas turbine, two features can change the composition of a homogeneous mixture: *mixing* with other mixtures, or *chemical reactions* occurring in the mixture. In a chemical reaction, some species disappear and new ones are formed. In practice a lot of chemical reactions take place, even in mixtures with a constant composition. Only in the case of a constant composition, the destruction of species is as big as the creation. When the destruction of species equals the creation, and thus the composition remains constant, a so-called state of *equilibrium* is reached. If this is the case, the thermodynamic properties of the mixture stay the same.

A lot of processes, like combustion, involve many reactions, with each reaction having its own speed, the so-called *reaction rate*. The reaction rate depends on the concentrations of the species involved, and the *specific reaction-rate constant*. The specific reaction-rate constant depends on the temperature and the activation energy, according to Arrhenius' law:

$$k = AT^{\beta} \exp\left(-\frac{E_a}{RT}\right) \quad (\text{B.1}),$$

where:

- k = specific reaction-rate constant ($\text{mole}^{1-m}\text{cm}^{3m-3}\text{s}^{-1}$),
- A = a constant depending on the reaction,
- β = a constant different per reaction (value is between 0 and 1),
- E_a = activation energy (J/mole).

The 'm' in the powers of the unity of 'k' denotes the order of the reaction, in other words the sum of the coefficients of the reactants in case of the forward specific reaction rate or the sum of the coefficients of the products in case of backward specific reaction rate.

For example, the reaction rate of the reaction



is:

$$\frac{dC_A}{dt} = -k \cdot C_A C_B = \frac{dC_B}{dt} = -\frac{dC_C}{dt} = -\frac{dC_D}{dt} \quad (\text{B.3}),$$

where:

- C_x = concentration of species x (mole/cm^3),
- t = time (s).

Because the arrow in reaction (B.2) only points to the right, (B.2) involves only the reaction forming C and D from A and B. This reaction is of second order, because the coefficients of A and B are both one, so the unity of k is ($\text{mole}^{-1}\text{cm}^3\text{s}^{-1}$). It is obvious from (B.3) that the reaction rate depends on the specific reaction-rate constant and the concentrations of the species. Suppose now that reaction (B.2) would be modified in that way that the backward reaction would also become important (i.e. A and B are also formed from C and D). Then the so-called ‘one-way equilibrium reaction rates’, mentioned in chapter 6, can be calculated. These are equal to the product of the forward specific reaction rate constant and the equilibrium reactant concentrations. Because equilibrium concentrations are used, this is also equal to the product of backward reaction rate constant and equilibrium product concentrations. For the modified reaction, the one-way equilibrium reaction rate is:

$$R_{B.2} = k_{B.2f} \cdot C_{A,eq} C_{B,eq} = k_{B.2b} \cdot C_{C,eq} C_{D,eq} \quad (\text{B.4}).$$

A set of reactions leading to the formation of certain products is called a *reaction scheme* (also reaction mechanism) or *kinetic scheme*. If the different reactions in the reaction scheme are added together, the *overall reaction* is found. The reaction rates of different reactions in a scheme can differ very much. If some reactions in the mechanism are a lot slower than the other reactions in the mechanism, they can limit the speed of the overall reaction.

To describe processes involving chemical reactions, a choice has to be made on what reaction scheme is to be used. A reaction scheme involving more reactions can give a more adequate vision of reality, but such a scheme is also more complex than shorter reaction schemes. In case of computer simulations, a complex reaction scheme can take a lot of computing time and memory.

If all the reactions in a certain reaction scheme are very fast, a state of equilibrium is reached in a short time. If this equilibrium involves disappearance of (almost) all the reactants, and formation of new products, an infinitely fast overall reaction can be acceptable. In a lot of problems comprising combustion, this approach is assumed accurate enough, as long as there is enough oxygen.

The question whether reactions are fast enough to assume the equilibrium state to be reached depends on the speed of competing processes taking place. Apart from the speed of reactions a *characteristic time* that the reaction takes can also be used. In a gas turbine the residence time (e.g. in the combustion chamber) can be compared with the characteristic time of chemical reactions. If the residence time is long compared to the characteristic time of the reaction the

assumption can be made that the products reach equilibrium. If the characteristic time of the reaction is (much) longer than the residence time, the composition can be assumed to remain unchanged, more often called ‘frozen’.

B.2 Dissociation

For chemical reactions to occur it is not necessary that several species are present to react together. At high temperatures (how high differs per specie), bonds between atoms can spontaneously dissolve. This process has several different names, like cracking, in the case of hydrocarbons in a cracking plant, pyrolysis or dissociation.

In a gas turbine, cracking or pyrolysis occurs only locally in hot zones of the combustion chamber, or elsewhere on a very limited scale, in the case of incomplete combustion, e.g. in case of a flame out. Dissociation however, becomes more and more important in the hot sections of the engine, because the maximum temperatures in gas turbines are increasing. In a gas turbine most of the time hydrocarbons are burned. The overall reaction of the complete combustion of a hydrocarbon can be described by the following formula:



Dissociation of CO_2 and H_2O can be neglected at temperatures lower than 1800 (K), but in case there are unusually high concentrations of species (like carbon dioxide), dissociation can become important at lower temperatures.

Because bonds are dissolved, the potential energy of the gases increases when dissociation appears and the (sensible) heat decreases. Therefore, temperatures found assuming dissociation are lower than temperatures found when dissociation is neglected. Pressure has effect on dissociation if the number of moles is increased because of dissociation. In that case, a higher pressure counteracts dissociation. If dissociation is opposed, less bonds will be dissolved and less heat converted. The temperature will be higher then.

B.3 Heats of reaction and heats of formation

In general, the heat associated with a chemical reaction is an indefinite quantity, depending on the path taken. However, if the chemical reaction is carried out at constant pressure or constant volume, the heat change has a definite value, determined solely by the initial and final states of the system.

An extension of this is known as Hess' law (also called the law of constant heat summation). This law states that the resultant heat change, at constant pressure or at constant volume, in a given chemical reaction is the same whether it takes place in one or several stages. This means that the net heat of reaction depends only on the initial and final states. So the heat of reaction can be calculated by simply adding thermochemical equations; one can choose a suitable reaction path.

The heat of reaction is defined as follows. If a closed system containing a number of moles at a given temperature and pressure undergoes an isobaric process to a prescribed composition having the same temperature as the system before the process, the heat liberated by the system is the heat of reaction for this process. It can be shown that for a reacting flow, with negligible change in potential and kinetic energy and no work done other than that required for flow, the heat of reaction is equal to the change in enthalpy. The first step in proving this is the first law of thermodynamics:

$$\Delta E = \delta Q - \delta W \quad (\text{B.6}),$$

where:

ΔE	= increase in energy content of the system,
δQ	= heat absorbed by the system,
δW	= work done by the system.

For the increase in energy content of the system one can also write:

$$\Delta E = \Delta U + \Delta PE + \Delta KE \quad (\text{B.7}),$$

where:

ΔU	= change in internal energy of the system,
ΔPE	= change in potential energy of the system,
ΔKE	= change in kinetic energy of the system.

If the changes in potential and kinetic energy of the system are negligible, combining equations (B.6) and (B.7) gives:

$$\Delta U = \delta Q - \delta W \quad (\text{B.8}).$$

If the reaction occurs at a constant pressure the following can be written:

$$Q_p = \int (dQ)_p = \Delta U + p\Delta V \quad (\text{B.9}).$$

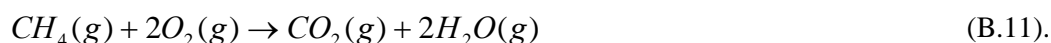
For a change from state A to state B:

$$Q_p = U_B - U_A + p(V_B - V_A) = H_B - H_A = \Delta H \quad (\text{B.10}).$$

Equation (B.10) shows that, under the given assumptions, the heat of reaction is equal to the change in enthalpy.

The heat of reaction should not be confused with the heat of formation. The standard heat of formation of a substance is defined as the heat evolved when one mole of a substance is formed from its elements in their respective standard states (standard temperature and pressure). For a reaction between species, the heat of reaction can be calculated from the heats of formation of the species using Hess' law.

As an example, the heat of reaction of methane reacting with oxygen in stoichiometric content will be calculated at standard state conditions. The assumption is made that the products formed are carbon dioxide and water. The overall reaction is:



Because the heat of formation of oxygen is zero at standard state conditions (oxygen is an element) only the heats of formation of the other three species are needed. The chemical reactions in which these species are formed from their elements are:



The heat of reaction can be found by first assuming the opposite of reaction (B.12) to take place. Now, C(s) and 2 H₂(g) are formed, and a heat of 74600(J/mole) is consumed. After that, reactions (B.13) and (B.14) are assumed to take place in the proportion 1:2 respectively. This time a heat of 393510+2*241826 = 877162 (J/mole) evolves. So the overall heat of reaction is found to be:

$$Q_p = 877162 - 74600 = 802562(\text{J} / \text{mole}) \quad (\text{B.15}).$$

In equation (B.15), moles of reactions are meant. It is also possible to calculate the heat of formation of a specie exactly in the opposite way from the heat of reaction if the heats of formation of the other products and reactants are known.

B.4 Calculating equilibrium compositions

In general, two calculation methods can be used to obtain equilibrium compositions at a given temperature and pressure. The first one is by assuming a number of chemical reactions and subsequently calculating the concentrations using the equilibrium constants for these reactions. Because the equilibrium state is reached, all reactions are in equilibrium and it doesn't matter which reactions are used (as long as the species assumed to be present in the equilibrium are participating in the reactions). The equilibrium constant is a relation between concentrations and the pressure, depending solely on the temperature. The general form of an equilibrium constant for a reaction:



is:

$$K_p(T) = \frac{p_C^c \cdot p_D^d}{p_A^a \cdot p_B^b} \quad (\text{B.17}).$$

Using Dalton's law for partial pressures,

$$\frac{p_i}{p} = \frac{n_i}{N} \quad (\text{B.18}),$$

where: n_i = number of moles of specie i,
 p_i = partial pressure of specie i,
 N = total number of moles in the complete mixture.

This equation can be changed into an equation comprising numbers of moles and pressure:

$$K_p(T) = \frac{n_C^c \cdot n_D^d}{n_A^a \cdot n_B^b} \cdot \left(\frac{p}{N} \right)^{c+d-a-b} \quad (\text{B.19}).$$

This equation is valid when no condensed species are present. For a mixture of species, a number of equilibrium constants can be used. Using the principle of conservation of atoms, additional equations are found. Combining these last equations with the equilibrium constants gives a system of equations, which have to be solved simultaneously to find the equilibrium composition.

The other way to calculate the equilibrium composition is by optimisation of a thermodynamic function. This optimisation can be a maximisation, in the case of entropy, or a minimisation, in the case of a free energy function. What free energy function to use depends on the thermodynamic data given. An important law in thermodynamics states that with two thermodynamic properties (pressure, temperature, entropy, etc.) given, the thermodynamic state of the mixture is fixed and can be calculated. If the thermodynamic state is characterised using temperature and pressure for example, Gibbs energy is most easily minimised inasmuch as temperature and pressure are its natural variables.

Because the thermodynamic properties can be calculated for each of the species present, the effect of changes in composition on the entropy or a free energy function of the mixture can be calculated. Finding the optimum value of the thermodynamic function gives the equilibrium composition.

Gordon and McBride (Ref. 15) argue that, if a generalised method of solution is used, both formulations reduce to the same number of iteration equations. An advantage of free energy minimisation is that the species can be treated separately without assumption of reaction equations. It should be reminded, however, that the reactions themselves do not matter, only the relation between the concentration and pressure does. Glassman (Ref. 14) states that for easy problems it is most convenient to deal with equilibrium constants. When the problems become more complex, free energy minimisation becomes more attractive, because using equilibrium constants results in more bookkeeping, more numerical difficulties with the use of components, more difficulty in testing for the presence of condensed species and more difficulty in extending the generalised methods to conditions that require non-ideal equations of state.

Therefore, if the reactions and species are not very numerous and reaction equations can easily be found, as is the case for dissociation with temperatures not exceeding 3000 (K), the equilibrium constants are considered a good way to find the equilibrium. If the number of species or reactions become too big, or if liquid species can become significant, the optimisation could be a better option.

Appendix C Detailed description of gas models

C.1 GSP 7.0

In GSP 7.0, the ideal gas law is used as the equation of state, relating pressure, density and temperature.

The specific heat at constant pressure

The specific heat at constant pressure for a mixture is calculated using a seventh degree polynomial function of T. To find this polynomial the program starts with two polynomials: the specific heat of air, c_{pa} , and the specific heat of the combustion products in the case of stoichiometric combustion, c_{pf} , both as a function of temperature. This last polynomial is given in the program. It could also be calculated by taking the weighted average of the specific heats of carbon dioxide and water in the ratio in which they are encountered in the stoichiometric combustion products. This ratio follows from the H/C-ratio of the fuel. Because this polynomial c_{pf} is given, the gas model only takes one H/C-ratio into account.

From the polynomials of air and stoichiometric combustion products one new polynomial is built: c_{ps} . This c_{ps} is used to find the specific heat at constant pressure of a mixture using the following formula:

$$c_{pg} = \frac{c_{pa} + FAR \cdot c_{ps}}{1 + FAR} \quad (C.1),$$

where: FAR = fuel-air-ratio.

c_{ps} can be calculated using the above formula for the case of a stoichiometric combustion. In that case, c_{pg} is equal to the above mentioned c_{pf} , and FAR is equal to FAR_{stoich} , the stoichiometric fuel-air-ratio. Inserting this into the equation above:

$$c_{pf} = \frac{c_{pa} + FAR_{stoich} \cdot c_{ps}}{1 + FAR_{stoich}} \quad (C.2),$$

In this equation only c_{ps} is unknown. Thus, c_{ps} can be found from:

$$c_{ps} = \frac{c_{pf} \cdot (1 + FAR_{stoich}) - c_{pa}}{FAR_{stoich}} \quad (C.3).$$

FAR_{stoich} is, like the H/C-ratio, fixed in the program. Because c_{pf} and c_{pa} are also fixed in the program, c_{ps} is also fixed and is only calculated one time in the program (during the

initialisation). To be perfectly clear c_{ps} is not the c_p of a really existing gas, it's just used to make the calculations simpler. The specific heat at constant pressure for a mixture as a function of temperature and fuel-air-ratio is found by using equation (C.1).

The enthalpy

The enthalpy is calculated in the same way as the specific heat at constant pressure. Here also seventh degree polynomials are used for the enthalpy of air (h_a) and stoichiometric combustion products (h_f), and from these two a polynomial h_s is made, according to the following formula, which has exactly the same form as equation (C.3):

$$h_s = \frac{h_f \cdot (1 + FAR_{stoich}) - h_a}{FAR_{stoich}} \quad (C.4).$$

Like c_{ps} , h_s is not the enthalpy of a real gas, it's only calculated as a matter of convenience. The enthalpy of a mixture as a function of temperature and fuel-air-ratio is calculated using:

$$h_g = \frac{h_a + FAR \cdot h_s}{1 + FAR} \quad (C.5).$$

The specific gas constant

The specific gas constant can generally be found by dividing the universal gas constant, R , by the molar mass of the mixture, M . In GSP 7.0, the specific gas constant is calculated by taking the weighted average between R_a , the value of the specific gas constant for air, which has a value of 287.057, and R_s , which has a value of 287.056. The specific gas constant is calculated according to the formula:

$$R_g = \frac{R_a + \frac{FAR}{FAR_{stoich}} \cdot [(1 + FAR_{stoich}) \cdot R_s - R_a]}{1 + FAR} \quad (C.6).$$

A strange thing about this formula is that it has exactly the same form as formula (C.1) with formula (C.3) inserted in it. For that reason it would be more logical if R_s would be R_f . At an H/C-ratio of about 1.95, the specific gas constant of the combustion gases is about the same as the specific gas constant of air.

The formula can be further investigated by inserting the values GSP uses for the constants in the formula. One finds:

$$R_g = \frac{287.057 + 287.0559 \cdot FAR}{1 + FAR} \quad (C.7).$$

From this formula it can clearly be seen that GSP always finds a value of approximately 287.057 ($\pm 7.1 \text{ e-5}$), as was to be expected.

The entropy

Unlike the enthalpy the entropy is not only a function of temperature, but also of pressure. The first step GSP 7.0 makes when calculating the entropy is creating a function Φ (phi). Φ is only a function of temperature and fuel-air-ratio. It is calculated in about the same way as enthalpy and specific heat at constant pressure: again two seventh degree polynomial functions of temperature, Φ_a and Φ_f are given in GSP 7.0. Using a formula exactly like (C.3) and (C.4) the function Φ_s is made.

A difference with the calculation of the enthalpy and the specific heat at constant pressure is that this time, one term is added to the polynomials of Φ_a and Φ_s , namely the specific heat at constant pressure multiplied by the natural logarithm of temperature. The specific heat at constant pressure is given as a constant here: $c_{pa} = 1021.10 \text{ (J/kg/K)}$. Using formula (C.3) and the following constant value for c_{pf} : $c_{pf} = 1032.57 \text{ (J/kg/K)}$, the value for c_{ps} becomes 1033.361 (J/kg/K). Now that the polynomials of Φ_s and Φ_a , including the logarithmic term, are known, the polynomial for Φ_g , the function Φ for a mixture, can be calculated using:

$$\Phi_g = \frac{\Phi_a + FAR \cdot \Phi_s}{1 + FAR} \quad (C.8).$$

If Φ_g is known for the mixture, it is only a small step to calculating the entropy. GSP 7.0 uses the following formula:

$$S_g = \Phi_g - R_g \cdot \ln \frac{p}{p_{std}} \quad (C.9),$$

where: p = pressure (Pa),
 p_{std} = standard pressure (here 101325 (Pa)).

The ratio of specific heats

To calculate the ratio of specific heats (the specific heat at constant pressure divided by the specific heat at constant volume) the specific heat at constant volume, c_v , must be known. For ideal gases c_v can be calculated as the difference between c_p and the specific gas constant R_g . Thus, in GSP 7.0 γ is calculated as follows:

$$\gamma = \frac{c_p}{c_p - R_g} \quad (C.10).$$

The speed of sound

To calculate the speed of sound, a , GSP 7.0 uses the formula:

$$a = \sqrt{\gamma R_g T} \quad (C.11).$$

If the temperature and fuel-air-ratio are known, γ and R_g can be calculated, and thereby the speed of sound.

The dynamic viscosity

The dynamic viscosity is not calculated in the GSP 7.0 gas model.

C.2 GasTurb 7.0

Like in GSP 7.0, the ideal gas law is used as the equation of state.

The specific heat at constant pressure

The specific heat at constant pressure, c_p , is calculated from a polynomial, dependent on the temperature. This polynomial is found from two other polynomials by using the fuel-air-ratio. The only difference with the calculation of the c_{pg} polynomial in GSP, is that here the equivalent of c_{ps} is given, and not calculated in the program. In GSP 7.0 this also could have been done, since c_{ps} was calculated from c_{pf} , c_{pa} and FAR_{stoich} , where FAR_{stoich} and the coefficients of c_{pa} and c_{pf} are all constants. The equivalent of c_{ps} is called c_{pf} . The specific heat at constant pressure is called c_{pl} (air is 'Luft' in German). The c_{pg} -polynomial is calculated from the following equation, the GasTurb 7.0 equivalent of equation (C.1):

$$c_{pg} = \frac{c_{pl} + FAR \cdot c_{pf}}{1 + FAR} \quad (C.12).$$

The enthalpy

The enthalpy is calculated in the same way as the c_p : there are two polynomials, h_l for air and h_f for a virtual gas, and the enthalpy polynomial of the mixture, h_g , follows from these two polynomials and the fuel-air-ratio:

$$h_g = \frac{h_l + FAR \cdot h_f}{1 + FAR} \quad (C.13).$$

A remark that can be made here is that for temperatures below 180 (K) the enthalpy is calculated with the formula:

$$h_g(T) = \frac{h_l(180) + FAR \cdot h_f(180)}{1 + FAR} \cdot \frac{T}{180} \quad (C.14).$$

Studying the formula, one can see that $h_g(T)$ is made a linear function between $T=0$ (K), where $h_g = 0$ (J/kg) and $T=180$ (K), where h_g is equal to the value calculated with equation (C.13).

The specific gas constant

For the specific gas constant, GasTurb 7.0 uses a constant value: $R_g = 287.05$ (J/kg/K).

According to GasTurb's user manual, a fixed H/C-ratio of 1.9405 is assumed. In that case, the specific gas constant of the reaction products is the same as for air.

The entropy

The absolute value of the entropy is nowhere calculated in GasTurb 7.0. Entropy is only needed to calculate isentropic changes. For that purpose, a function $\Phi(T)$ is defined, using the same coefficients as the polynomials for specific heat at constant pressure and the enthalpy: the only difference is a constant integration factor. The definition of Φ is:

$$\Phi(T) = \frac{1}{R_g} \cdot \int_0^T \frac{c_p(T)}{T} dT \quad (C.15).$$

By using this formula for $\Phi(T)$ the following equation can be found for an isentropic change of state:

$$\ln\left(\frac{p_2}{p_1}\right) = \Phi_2(T) - \Phi_1(T) \quad (C.16).$$

Studying this function $\Phi(T)$ points out that it is the same one as used in GSP 7.0, except for the factor R_g , the specific gas constant. If equation (C.9) is used for 2 states with the same value for the entropy, one finds:

$$\Phi_2(T) - \Phi_1(T) = R_g \left\{ \ln\left(\frac{p_2}{p_{std}}\right) - \ln\left(\frac{p_1}{p_{std}}\right) \right\} = R_g \ln\left(\frac{p_2}{p_1}\right) \quad (C.17).$$

It is obvious that except for the factor R_g , equations (C.16) and (C.17) are the same. The $\Phi(T)$ -polynomial is calculated from the polynomials $\Phi_l(T)$, the polynomial for air, and $\Phi_f(T)$, the polynomial for a virtual mixture. Like for the enthalpy, GasTurb 7.0 contains an approximation for $\Phi(T)$ for temperatures below 180 (K): the program calculates the values of Φ for 180 (K) and 185 (K), and draws a straight line through those points. The values for temperatures below 180 (K) are found by extrapolating this line. For temperatures above 2300 (K) the same method of extrapolation is used, using values for Φ at 2300 (K) and 2305 (K).

The ratio of specific heats

The ratio of the specific heats (γ) is calculated in exactly the same way as in GSP 7.0:

$$\gamma = \frac{c_p}{c_p - R_g} \quad (C.18).$$

The speed of sound

The speed of sound is not calculated in a separate procedure, but calculated where needed using the formula:

$$a = \sqrt{\gamma(T_s) \cdot R_g T_s} \quad (C.19).$$

The subscript s in T_s indicates that static temperatures are used here.

The dynamic viscosity

Finally, the dynamic viscosity is calculated using two fourth degree polynomials, functions of the temperature. The first polynomial is the dynamic viscosity as a function of temperature for air only (μ_1 in the equation below), and the second is the dynamic viscosity for a mixture of air with combustion products that exist when the fuel-air-ratio is 0.04 (μ_2 in the equation below). The dynamic viscosity of the mixtures at other fuel-air-ratios is found by assuming a linear dependence of the dynamic viscosity on fuel-air-ratio:

$$\mu = \mu_1(T) + \frac{FAR}{0.04} (\mu_2(T) - \mu_1(T)) \quad (C.20).$$

C.3 GasTurb 8.0

As was said in the introduction, GasTurb 8.0 offers users the possibility to choose one of several different fuels. The heating value of the fuel is fixed, except for standard fuel, which also has a

default value for the fuel heating value. Standard fuel is the jet fuel that was also used in version 7.0.

The calculation of thermodynamic properties of the mixture when standard fuel is used is the same as in version 7.0. When a different fuel is specified, the values of thermodynamic properties are found by interpolation in tables. This interpolation is linear, except for the function Φ . Here logarithmic interpolation is used. Because most thermodynamic properties are a function of fuel-air-ratio and temperature, most of the tables are two-dimensional. This is the case for specific heat at constant pressure, ratio of specific heats, enthalpy and the function Φ (used when calculating isentropic changes). The molar mass is found in a one-dimensional table because it only depends on the composition (here fuel-air-ratio). As long as dissociation is neglected, the temperature has no significant effect on composition, which means that fuel-air-ratio is enough to predict the molar mass of a mixture. The specific gas constant is calculated by dividing the universal gas constant by the molar mass:

$$R_g = \frac{R}{M} \quad (\text{C.21}),$$

where: M = Molar mass (g/mole).

C.4 NASA CEA-Program

C.4.1 Calculation of thermodynamic and thermal transport properties

In CEA, the perfect gas law is assumed to be valid even when small amounts of condensed species (up to several percent by weight) are present. Because condensed species are assumed to occupy a negligible volume relative to the gaseous species, volumes used in CEA refer to gases only, while mass includes condensed species.

The CEA-program contains a lot of data in polynomial form for solid, liquid and gaseous species, relevant to application in the in chapter three mentioned type of problems (e.g. rocket performance). The ranges of the polynomials are from 200 (K) to 1000 (K), from 1000 (K) to 6000 (K) and from 6000 (K) to 20000 (K), as long as the specie considered can still be present in relevant (equilibrium) quantities at that temperature. The polynomials for thermodynamic properties are given per specie in the form of ten least-squares coefficients (a_1 up to and including a_8 , b_1 and b_2). For some (less important) species less coefficients are used to describe the polynomial. For thermal transport properties, four least-squares coefficients are used. The notations used here can differ from NASA's notations.

The specific heat at constant pressure

The specific heat at constant pressure per specie is calculated by means of a polynomial. To make the polynomial dimensionless, it is given in the form of (c_p/R):

$$\frac{c_p(T)}{R} = a_1 T^{-2} + a_2 T^{-1} + a_3 + a_4 T + a_5 T^2 + a_6 T^3 + a_7 T^4 \quad (\text{C.22}).$$

The c_p found using this equation is in (J/mole/K). To get c_p in (J/kg/K), one can use the molar weight of the specie. The c_p of a mixture is calculated by taking the weighted average between the values of c_p of the species present in the mixture. In case of a c_p in (J/kg/K) the mass fraction is to be used and in case of c_p in (J/mole/K) the mole fraction:

$$c_{pg} = \sum_{i=1}^{NS} (m_i \cdot c_{p,i}) \quad (\text{C.23}),$$

where:

- c_{pg} = specific heat at constant pressure of the (gaseous) medium (J/kg/K),
- NS = number of species in mixture (-),
- m_i = mass fraction of specie i (-),
- $c_{p,i}$ = specific heat at constant pressure of specie i (J/kg/K).

The c_p calculated using equation (C.23) is the c_p of a mixture, assuming that the composition remains constant ('frozen' composition, see appendix on chemical reactions). When the assumption is made that reactions can occur during changes of temperature, another c_p should be found. For example, if condensed species are present an increase in temperature can cause evaporation. Because of evaporation, more heat will be needed to increase the temperature of the mixture, and consequently, a larger c_p is found, than is the case when not taking evaporation into account. Therefore, NASA discerns the $c_{p,e}$, with the assumption of equilibrium, and the $c_{p,f}$, using the assumption of frozen compositions. The difference between these two is called $c_{p,r}$, the c_p change due to reactions. The $c_{p,r}$ is calculated iteratively together with the iterations used to find the equilibrium composition, see paragraph C.4.2, using the following formula:

$$c_{p,r} = \sum_{j=1}^{NG} n_j \frac{h_j}{T} \left(\frac{\partial \ln n_j}{\partial \ln T} \right)_p + \sum_{j=NG+1}^{NS} \frac{h_j}{T} \left(\frac{\partial n_j}{\partial \ln T} \right)_p \quad (\text{C.24}),$$

where:

- n_j = number of moles of specie j per gram of mixture,
- h_j = enthalpy of specie j,
- NS = number of species,

NG = number of gases.

Remark: in the NASA-program, gases are indexed from 1 to NG and condensed species are indexed from $NG+1$ to NS .

The enthalpy

The enthalpy is also calculated per specie with a polynomial, partly using the same coefficients as used for the specific heat at constant pressure. The enthalpy is also divided by the (specific) gas constant. The polynomial is:

$$\frac{h(T)}{R} = -a_1 T^{-1} + a_2 \ln T + a_3 T + a_4 \frac{T^2}{2} + a_5 \frac{T^3}{3} + a_6 \frac{T^4}{4} + a_7 \frac{T^5}{5} + b_1 \quad (C.25).$$

The value of b_1 can be chosen arbitrarily, but the slope of $h(T)$ is determined by the specific heat at constant pressure. In the NASA-program the value of b_1 is determined by the condition, that the value of the enthalpy at a temperature of 298.15 (K) is equal to the heat of formation of that specie at 298.15 (K). Again, the enthalpy for the mixture is found using:

$$h_g = \sum_{i=1}^{NS} (m_i \cdot h_i) \quad (C.26),$$

where:

h_g	= enthalpy of the medium (J/kg),
NS	= number of species in mixture (-),
m_i	= mass fraction of specie i (-),
h_i	= enthalpy of specie i (J/kg).

The specific gas constant

The specific gas constant is not used explicitly in the NASA-program, only the universal gas constant is used as well as the molar mass. NASA uses the following expression for the ideal gas law:

$$\frac{p}{\rho} = nRT \quad (C.27),$$

where: n = total number of moles of *gaseous* species per gram mixture.

It is obvious that 'nR' in the equation is the same as the specific gas constant. The value for 'n' is found using:

$$n = \sum_{j=1}^{NG} n_j \quad (C.28),$$

where: n_j = number of moles of specie j per gram mixture.

For the molar mass, two definitions are used, the first one is:

$$M = \frac{\sum_{j=1}^{NS} n_j M_j}{\sum_{j=1}^{NG} n_j} = \frac{1}{n} \quad (C.29).$$

The other definition of molar mass is:

$$MW = \frac{\sum_{j=1}^{NS} n_j M_j}{\sum_{j=1}^{NS} n_j} \quad (C.30).$$

The relationship between these two definitions is:

$$MW = M \left(1 - \sum_{j=NG+1}^{NS} x_j \right) \quad (C.31),$$

where: x_j = mole fraction of condensed specie j relative to all species.

Obviously, the specific gas constant is only equal to the universal gas constant divided by the molar mass if no condensed species are present. In that case, both definitions of the molar mass yield the same result.

The entropy

The entropy is partly calculated using a polynomial. The entropy per specie follows from:

$$S_j = S_j^0 - R \ln \frac{n_j}{n} - R \ln p \quad (j=1, \dots, NG) \quad (C.32),$$

$$S_j = S_j^0 \quad (j=NG+1, \dots, NS) \quad (C.33),$$

where: S_j^0 = standard state molar entropy for species j.

Equation (C.32) is only valid for gases, and equation (C.33) for condensed species. The standard state molar entropy for species j follows from a polynomial, partly using the same coefficients as the polynomials for specific heat at constant pressure and the enthalpy:

$$\frac{S_j^0(T)}{R} = -a_1 \frac{T^{-2}}{2} - a_2 T^{-1} + a_3 \ln T + a_4 T + a_5 \frac{T^2}{2} + a_6 \frac{T^3}{3} + a_7 \frac{T^4}{4} + b_2 \quad (\text{C.34}).$$

The total number of moles of gaseous species per gram mixture (n) in equation (C.32) follows from equation (C.28).

The ratio of specific heats

In the NASA-program an explicit difference is made between the ratio of specific heats, γ , and another thermodynamic property, γ_s , used to calculate the speed of sound. The definition of both is:

$$\gamma_s \equiv \left(\frac{\partial \ln p}{\partial \ln \rho} \right)_s \quad (\text{C.35}),$$

$$\gamma \equiv \frac{c_p}{c_v} \quad (\text{C.36}).$$

The relationship between both is:

$$\gamma_s = - \frac{\gamma}{\left(\frac{\partial \ln V}{\partial \ln p} \right)_T} \quad (\text{C.37}).$$

It can be shown that for an ideal gas:

$$\left(\frac{\partial \ln V}{\partial \ln p} \right)_T = -1 \quad (\text{C.38}).$$

So if the gas behaves like an ideal gas, γ_s and γ are the same.

The speed of sound

The speed of sound is found from the following equation:

$$a = \sqrt{nRT\gamma_s} \quad (\text{C.39}).$$

In this equation, n is calculated using equation (C.28).

The dynamic viscosity

CEA can calculate several thermal transport properties. These are the dynamic viscosity, the thermal conductivity and the Prandtl number. Because only the dynamic viscosity is used in the new gas model, this is the only thermal transport property whose calculation is described here. The dynamic viscosity is only calculated for gaseous species, because there is no feasible method for calculating thermal transport properties for a multiphase mixture.

The dynamic viscosity per specie is calculated as a function of temperature using a polynomial, with least-squares coefficients (A, B, C and D):

$$\ln(\mu) = A \ln(T) + \frac{B}{T} + \frac{C}{T^2} + D \quad (\text{C.40}).$$

The unity of dynamic viscosity used by CEA is the micropoise (μP). The dynamic viscosity of a mixture cannot be calculated by simply taking the weighted average of the dynamic viscosity of the species, because interactions between the species also influence the dynamic viscosity. To find the dynamic viscosity for a mixture, the following formula is used:

$$\mu_{\text{mix}} = \sum_{i=1}^{NG} \frac{x_i \mu_i}{x_i + \sum_{j=1}^{NG} x_j \phi_{ij}} \quad (\text{C.41}),$$

where:

NG	= number of gaseous species in the mixture (-),
x_i	= mole fraction of specie i relative to all <i>gaseous</i> species (-),
μ_i	= dynamic viscosity for specie (μP),
ϕ_{ij}	= viscosity interaction coefficient between species i and j (-).

In general, $\phi_{ij} \neq \phi_{ji}$. The interaction coefficient can be calculated by two equations:

$$\phi_{ij} = \frac{\mu_i}{\mu_{ij}} \frac{2M_j}{M_i + M_j} \quad (\text{C.42}),$$

where:

μ_{ij}	= viscosity interaction parameter (μP),
M_i	= molar mass of specie i (g/mole), and

$$\phi_{ij} = \frac{1}{4} \left\{ 1 + \sqrt{\frac{\mu_i}{\mu_j} \left(\frac{M_j}{M_i} \right)^{1/4}} \right\}^2 \sqrt{\frac{2M_j}{M_i + M_j}} \quad (\text{C.43}).$$

Equation (C.42) can be used only if the viscosity interaction parameter between species is known. If this is not the case, equation (C.43) is used.

C.4.2 Calculation of equilibrium compositions

The CEA-program can calculate equilibrium compositions if one of six combinations of two thermodynamic state functions is specified. The program finds the equilibrium composition by optimisation of a thermodynamic function, as described in appendix B. For combustion at constant pressure, which is approximately the case in gas turbines, the two thermodynamic state functions are enthalpy and pressure. In this case, Gibbs energy is minimised by CEA. The enthalpy follows from the reactant temperature and composition, by using the heats of formation.

For more detailed information on the optimisation procedure one should read Gordon (Ref. 15).

C.5 A constant specific heat gas model

An example of such a gas model is the following. Two different gas streams are discerned: air and air with combustion products. The following properties of the gases are used:

Table C.1 Example of a constant specific heat gas model

Properties:	Air	Comb. Prod.
Specific gas constant, R_g (J/kg/K)	287	287
Specific heat at constant pressure, c_p (J/kg/K)	1000	1150
Ratio of specific heats, γ (-)	1.4	1.33

The enthalpy is found by multiplying the constant specific heat at constant pressure with temperature and adding a(n) (arbitrary) constant. Entropy is not explicitly calculated in this model, but for an isentropic change of state, the Poisson equation can be used to relate changes in temperature and pressure:

$$\frac{T_2}{T_1} = \left(\frac{p_2}{p_1} \right)^{\frac{\gamma-1}{\gamma}} \quad (\text{C.44}).$$

The speed of sound is calculated using:

$$a = \sqrt{\gamma R_g T} \quad (\text{C.11}).$$

where: a = speed of sound (m/s),
 R_g = specific gas constant (J/kg/K).

For the equation of state, the ideal gas law is used. In such an easy gas model, usually no thermal transport properties are calculated, because they are less important for gas turbine performance prediction than the thermodynamic properties.

Appendix D Detailed description of combustor models

D.1 GSP 7.0

The combustion chamber is modelled as a black box in GSP 7.0. The only combustion process that can be described by GSP 7.0 is complete (infinitely fast) combustion of a fuel. Only one fuel property can change: the fuel heating value. In the program, a scale factor is calculated by dividing the user specified fuel heating value by the standard fuel heating value. This scale factor is used within the program.

However, within the program the fuel heating value is not a number, but a decreasing polynomial function of the (burner exit) temperature. To understand why this function is decreasing, one should take a look at figure J.3, and equation (J.1), repeated here for convenience:

$$H_{\text{reactants}} + \Delta H_r(T_1) = \Delta H_{\text{products}} + \Delta H_r(T_2)$$

When the products of the (complete) combustion reaction are assumed to be carbon dioxide and water, the heating value is equal to the heat of reaction. From equation (J.1) it is obvious that the difference between the heating values at two temperatures is formed by the difference in enthalpy changes between heating the reactants or products. The heating value is decreasing because enthalpy needed for the heating of the combustion products is lower than the enthalpy needed to heat the fuels.

However, the polynomial is not exactly equal to the heating value. This can be seen by studying the following equation, taken from GSP 7.0:

$$W_f \cdot FHV(T_{out}) \cdot \eta_{comb} = W_{in} \cdot H_{air}(T_{out}) - W_{in} \cdot H_{air}(T_{in}) \quad (D.1),$$

where:

- W_f = fuel flow (kg/s),
- FHV = fuel heating value (J/kg),
- η_{comb} = combustion efficiency (-),
- W_{in} = oxidant mass flow entering the combustion chamber (kg/s),
- H_{air} = enthalpy of the air only (J/kg).

This equation can be seen as a sort of definition of the fuel heating value. The left-hand term of equation (D.1) is (a sort of) heat release due to combustion, and the right-hand term is a sort of warming of the oxidant. The meaning of the second term on the right-hand side is quite clear: this is the total quantity of enthalpy (not per unit of mass) of the air entering the combustor. The first term, however, consists of the enthalpy per unit of mass of the air leaving the combustion chamber multiplied by the oxidant mass flow *entering* the combustion chamber. The true

amount of oxidant leaving the combustion chamber is of course less than W_{in} because some oxidant is taken away by the combustion process.

The choice to alter the 'fuel heating value polynomial' is obviously made to facilitate calculations. However, the price paid for this calculation ease is that the program code becomes less clear.

D.2 GasTurb 7.0

The combustor model is a black box. Like in GSP 7.0, GasTurb 6.0 used a polynomial for the (user-specified) fuel heating value. In GasTurb 7.0 (and 8.0) however, this polynomial isn't used anymore, because GasTurb 7.0 (and 8.0) takes dissociation into account. To find the burner exit temperature, a special procedure is used. This procedure uses a spline-interpolation in tables.

For a reference pressure the temperature rise over the combustion chamber is calculated as a function of the burner inlet temperature and fuel-air-ratio. If the pressure in the combustion chamber is not the same as the reference pressure, a correction factor is used, which is again a function of burner inlet temperature and fuel-air-ratio. The tabulated temperatures were found with a special computer program calculating equilibrium compositions and temperatures (such as the NASA-program).

Although GasTurb calculates burner exit temperatures accounting for dissociation effects, the program has its shortcomings concerning dissociation, because no dissociation corrections are performed in other components. As stated in the appendix on chemical reactions, dissociation takes away heat, and converts it into chemical energy. Because the heat decreases, the temperature found while accounting for dissociation will be lower than the temperature found when dissociation is neglected. When the temperature decreases in a process after the combustion has taken place, e.g. in an expansion process, there will be recombination of the previously dissociated species. In this process, chemical energy is converted back into heat, which will cause an increase in temperature. For this reason, it is not correct, to account for the effects of dissociation only in the combustion chamber. If dissociation occurs in the combustion chamber, some enthalpy is taken away by dissociation. Later on, in the turbine, recombination will give back enthalpy, but this effect is not accounted for in the program, so that enthalpy is lost.

D.3 GasTurb 8.0

In GasTurb 8.0 the burner exit temperature is basically calculated in the same way as in GasTurb 7.0, but now tables are added for the new fuels.

Appendix E Formation of emissions

E.1 Formation of UHC, CO and Smoke

Generally speaking, UHC (unburned hydrocarbons), CO and smoke are caused by incomplete burning of the fuel. This is probably caused by shortage of oxidant (normally oxygen (O_2)) or by low temperatures in certain regions of the combustion chamber. To avoid this problem the mixing process should be improved.

Unburned hydrocarbons are usually caused by a poor atomisation or a sudden drop of temperature, e.g. when the fuel reaches the film of cooling air, that is used to protect the material of the combustion chamber from the high temperatures. Lower amounts of unburned hydrocarbons can be achieved by proper injection and atomisation of the fuel, and temperatures in the burning zone high enough to sustain burning.

Carbon monoxide is formed by partial oxidation of hydrocarbons, and is therefore encountered in every combustion process involving carbon. Normally CO reacts with oxygen to form carbon dioxide (CO_2). CO can remain present in fuel-rich zones, because of a lack of oxygen, but also in fuel-lean zones because of dissociation of CO_2 . If the temperature or the radical concentrations (especially the OH concentration) become too low, the CO oxidation reactions, forming CO_2 , can be quenched. This quenching is likely to occur near the combustion chamber liner, because of the low temperature here. CO-emissions can be lowered by avoiding too rich and too lean combustion zones and providing a good burn-out with enough oxygen.

Smoke consists of carbon-rich particles. If a liquid fuel is used these particles can be formed when the residence time in the primary zone of the combustion chamber is too short for the fuel droplets to evaporate and burn completely. The amount of smoke can be reduced by avoiding (locally) too fuel-rich zones and by using a good atomiser to provide a good mixing process.

E.2 Formation of Nitrogen oxides

The formation mechanisms described here are mainly taken from (Glassman, Ref. 14), (Miller, Ref. 32) and (Michaud, Ref. 31).

NO_x is usually defined as the sum of NO and NO_2 , although sometimes also other nitrogen oxides like N_2O are included.

In combustion processes, NO is far more important than NO_2 . NO_2 is only encountered in the low-temperature regions of the combustion chamber. In the low-temperature regions of flames, the concentrations of HO_2 can be high enough to cause NO_2 formation from NO:



However, this NO_2 is rapidly removed by the following reactions:



Significant amounts of NO_2 can remain if the above reactions (E.2) and (E.3) quench because of decreasing radical concentrations due to mixing with cold fluid elements. More NO_2 is formed in the atmosphere (and possibly in the dilution zone) at even lower temperatures, because of the reaction:



The amounts of N_2O emitted into the atmosphere are negligible in high-temperature combustion processes, like the ones occurring in gas turbines. According to Kelsall (Ref. 25) this is even the case if low-calorific-value fuels are burned, with a (relatively) low adiabatic flame temperature. However, N_2O can be an intermediary product in the formation of NO , and thus contribute to NO_x formation.

Nitrogen oxides can be formed by four mechanisms:

- Thermal NO_x ,
- Prompt NO_x ,
- Fuel NO_x ,
- Nitrous oxides.

Thermal NO_x

Thermal NO_x can be described by the reaction mechanism proposed by Zeldovich:



Usually, this two-reaction mechanism is supplemented with a third reaction equation:



In table E.1 values for the constants in the equation for the specific reaction-rate constant are shown. This equation is shown in appendix B and repeated here for convenience:

$$k = AT^{\beta} \exp\left(-\frac{E_a}{RT}\right) \quad (\text{E.8}),$$

where: A, β = constants differing for each equation (-),
 T = temperature (K),
 E_a = activation energy (J),
 R = universal gas constant (8314.51 (J/mole/K)).

The reactions are repeated in table E.1, and the same numbers are used as written above. The only difference here is that the forward reaction and backward reaction are treated separately. In the reaction numbers, the 'f' stands for forward, and the 'b' stands for backward.

The value of the quotient (E_a/R) is denoted in the table as T_A .

Table E.1 Reaction constants for thermal NO_x-formation (Peeters, Ref. 35)

Reaction		A	β	T_A	k_{1000}	k_{1500}	k_{2000}	k_{2500}
E.5f	$\text{N}_2 + \text{O} \rightarrow \text{NO} + \text{N}$	$1.80 \cdot 10^{11}$	0	38367	$3.91 \cdot 10^{-6}$	$1.40 \cdot 10^0$	$8.39 \cdot 10^2$	$3.89 \cdot 10^4$
E.5b	$\text{NO} + \text{N} \rightarrow \text{N}_2 + \text{O}$	$4.01 \cdot 10^{10}$	0	508	$2.41 \cdot 10^{10}$	$2.86 \cdot 10^{10}$	$3.11 \cdot 10^{10}$	$3.27 \cdot 10^{10}$
E.6f	$\text{N} + \text{O}_2 \rightarrow \text{NO} + \text{O}$	$6.40 \cdot 10^6$	1	3139	$2.77 \cdot 10^8$	$1.18 \cdot 10^9$	$2.66 \cdot 10^9$	$4.56 \cdot 10^9$
E.6b	$\text{NO} + \text{O} \rightarrow \text{N} + \text{O}_2$	$1.37 \cdot 10^6$	1	19239	$6.04 \cdot 10^0$	$5.53 \cdot 10^3$	$1.82 \cdot 10^5$	$1.56 \cdot 10^6$
E.7f	$\text{N} + \text{OH} \rightarrow \text{NO} + \text{H}$	$3.00 \cdot 10^{10}$	0	0	$3.00 \cdot 10^{10}$	$3.00 \cdot 10^{10}$	$3.00 \cdot 10^{10}$	$3.00 \cdot 10^{10}$
E.7b	$\text{NO} + \text{H} \rightarrow \text{N} + \text{OH}$	$8.12 \cdot 10^{10}$	0	24125	$2.71 \cdot 10^0$	$8.41 \cdot 10^3$	$4.69 \cdot 10^5$	$5.23 \cdot 10^6$

Studying the table, it can clearly be seen that reactions E.5f, E.6b and E.7b depend strongly on temperature. The limiting step in the mechanism is E.5f, because of its high activation-energy. From the fact that E.5f is the limiting step for thermal NO_x-formation, the conclusion can be drawn that a high content of oxygen radicals promotes the formation of thermal NO_x, because it will increase the reaction rate for equation E.5f (see appendix B for the calculation of reaction rates).

In practice, the first two reactions from the Zeldovich mechanism ((E.5) and (E.6)) are far more important than the third one. However, the third reaction gets more significant in fuel-rich conditions.

Prompt NO_x

If one measures the build up of NO_x in a flame as a function of the time and extrapolates the amount of NO_x to time zero, a positive intercept on the NO_x -axis can be seen. This is called prompt NO_x . The time available in the flame front is too short for the Zeldovich mechanism to form this NO_x , assuming the equilibrium concentration of O radicals. According to Glassman (Ref. 14), three mechanisms can be responsible for formation of prompt- NO_x .

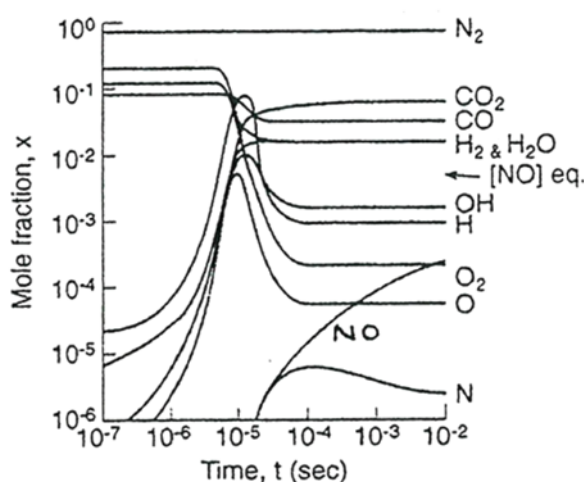


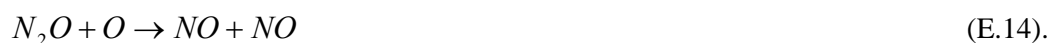
Figure E.1 Concentrations calculated for methane-air reaction (Glassman, Ref. 14)

The first mechanism is the Zeldovich mechanism, which is accelerated because of O and OH radical concentrations higher than equilibrium values. These O and OH concentrations are shown in figure E.1 for a stoichiometric methane-air reaction (at a pressure of 2 atmospheres and a temperature of 2477 (K)).

The second mechanism involves hydrocarbon fragments and atmospheric nitrogen. These two cause nitrogen-containing radicals, that are subsequently oxidised to NO_x . Examples of this sort of reactions are the following:



The third mechanism assumes the reaction of O-atoms with N_2 forming N_2O , which subsequently reacts to NO with another O-atom:



The first mechanism, involving O and OH concentrations above the equilibrium value, can be important in non-premixed flames, in stirred reactors for lean conditions and in low-pressure premixed flames. However, Miller argues that super-equilibrium O-atom concentrations are not very important as a source of prompt NO_x in flames, because the temperatures at which they occur are too low for the Zeldovich mechanism. The second mechanism, involving hydrocarbon fragments, is dominant in fuel-rich premixed hydrocarbon combustion and in hydrocarbon diffusion flames. The third mechanism, involving N_2O , becomes more important as the fuel-air-ratio decreases, the burned-gas temperature decreases, and as the pressure increases. However, it is most important under conditions where the total NO formation is relatively low.

Fuel NO_x

When the fuel used contains elemental nitrogen, a part of this nitrogen is converted to NO_x in the flame. According to Toof (Ref. 42), hydrogen cyanide is a necessary intermediate in the conversion of fuel bound nitrogen to NO_x . He describes the gross characteristics of the reaction mechanism for the formation of fuel NO_x as follows. In the first step hydrogen cyanide (HCN) is formed by pyrolysis of nitrogen-containing fuel. In the second step, amines (NH_i) are formed and in the third step a certain quantity of amines is oxidised to NO. Once NO is formed, the amines can react with NO as well as oxygen:



In equation (E.15) 'Ox' is a specie containing oxygen. The amount of nitrogen in the fuel that is converted to NO_x depends on the reaction rates of (E.15) and (E.16).

However, Miller (Ref. 32) claims that both hydrogen cyanide and ammonia (NH_3) can be intermediates in fuel NO_x formation. First the parent fuel nitrogen is converted rapidly to hydrogen cyanide and ammonia. Hydrogen cyanide is the most important product if the fuel nitrogen is bound in an aromatic ring whereas ammonia is the most important product if the fuel contains nitrogen in the form of amines. Both hydrogen cyanide and ammonia are subsequently oxidised to NO or N_2 , where amines form an important intermediate in the oxidation of hydrogen cyanide.

Assuming that Toof, who used older data than Miller, only used studies where fuel contained nitrogen was not present in the form of amines, the difference is not very big.

The reaction rate of the conversion of fuel bound nitrogen to fuel NO_x is usually high. An explanation for this is that the reaction mechanism uses a number of (rapid) steps that are also used by the (second) reaction mechanism forming prompt NO_x using hydrocarbon fragments.

The fraction of fuel bound nitrogen that is converted to NO, the so-called conversion rate, depends strongly on the local combustion environment (e.g. temperature and equivalence ratio), and on the initial level of nitrogen compound in the fuel-air mixture, but hardly on the molecule containing the nitrogen (e.g. pyridine, ammonia). In general, increasing the amount of fuel-bound nitrogen will result in a decrease of the conversion fraction.

Nitrous oxides

Nitrous oxide can be an important intermediate in NO formation and removal. In the literature, a number of possible steps involving N_2O have been mentioned, of which the most important are (M is a third specie):



Which reactions are important and whether the reactions form or remove N_2O depends on the combustion process. As an example, Miller (Ref. 32) claims that the most important reactions in a well-stirred reactor for methane-air combustion are the following. For lean and moderately rich fuel-air mixtures (equivalence ratio smaller than 1.2), the backward reaction of (E.17) is the principal N_2O formation reaction. The backward reaction of (E.22) starts forming N_2O around stoichiometric and for equivalence ratios higher than 1.2, the backward reactions of (E.22) and (E.21) become the most important N_2O formation steps. For all equivalence ratios encountered in the investigation (between 0.70 and 1.60), (E.20) is the prime N_2O removal reaction. For equivalence ratios lower than 1.2, reactions (E.21) and to a lesser extend (E.19) remove N_2O , while forming NO.

In general (like in the case of prompt NO_x formation by N_2O reactions), NO_x formation involving N_2O becomes more important as the fuel-air-ratio decreases, the burned-gas temperature decreases, and as the pressure increases. It is most important under conditions where the total NO formation is relatively low.

The relative contributions of the NO_x mechanisms depend on the combustion conditions, primarily the temperature and the equivalence ratio, and the fuel applied.

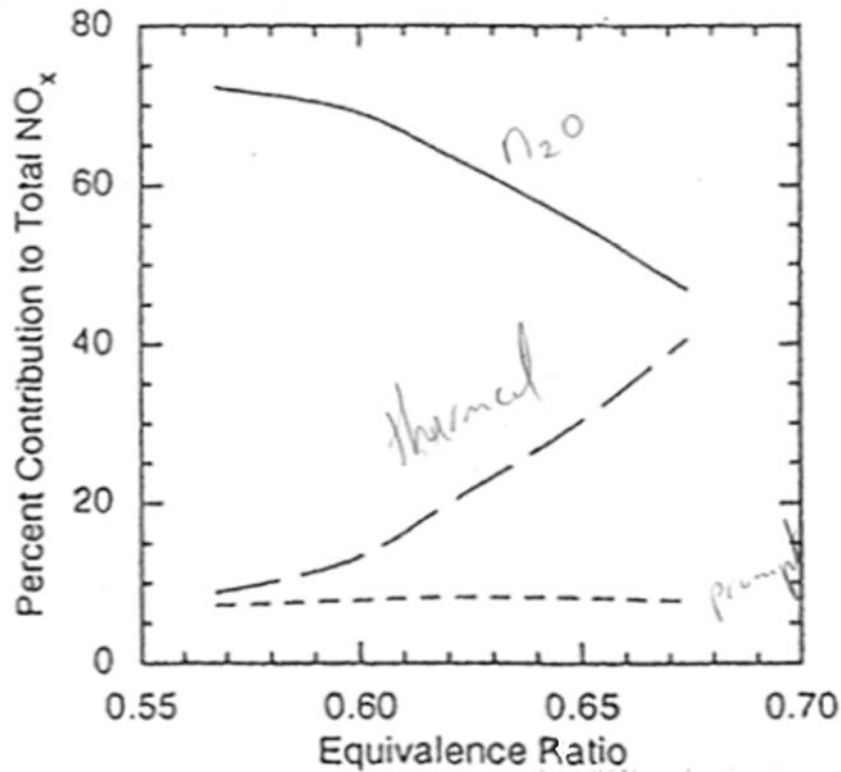


Figure E.2 Relative contributions of thermal, nitrous oxide and prompt NO_x to total NO_x (Michaud, Ref. 31)

At higher temperatures and equivalence ratios between 0.8 and 1, where most NO_x is formed, thermal NO_x is the most important source of NO_x -production. If the equivalence ratio gets lower than 0.8, the nitrous oxides pathway to NO_x becomes more important. In figure E.2 (Michaud, Ref. 31), the contributions of three mechanisms to total NO_x for a premixed methane-air flame are shown as a function of equivalence ratio for fuel-lean conditions. The solid line denotes nitrous oxide contribution, the dashed climbing line thermal NO_x contribution and the lower dashed line the prompt NO_x contribution. The contribution of prompt NO_x to total NO_x is relatively small under fuel-lean conditions (equivalence ratios smaller than 1). In case of fuel-rich combustion, however, prompt NO_x becomes relatively more important. Fuel NO_x is only present if the fuel contains nitrogen atoms bounded to other atoms than nitrogen (e.g. C or H).

Appendix F NO_x-reductions in gas turbines

As described in appendix E, there are four formation mechanisms by which NO_x-emissions are formed. When trying to reduce these emissions, the first thing one can think of is changing the fuel composition. By choosing a fuel without nitrogen the fuel-NO_x is zero. If a fuel without hydrocarbons would be used, prompt NO_x could also be reduced to low levels.

However, if total NO_x emission levels are high, by far the most important source of NO_x is thermal NO_x. Thermal NO_x can be described by the Zeldovich mechanism (see appendix E). The initiating reaction in the Zeldovich mechanism is very strongly dependent on temperature and O radical concentration. The most important way to reduce NO_x emissions is by lowering the highest temperatures in the gas turbine. These temperatures normally occur in the combustion chamber and the afterburner.

There are a number of ways to reduce the highest temperatures. The first way is by injecting liquid water or steam in the combustion chamber. Because the specific heat at constant pressure of water is quite high, a lot of enthalpy is needed to warm the water or steam, which causes a lower combustion temperature and thereby reduces the amount of thermal NO_x. This method is often applied in industrial gas turbines. For aircraft gas turbines, steam/water injection is hardly ever used because of the weight and volume of the water/steam that would have to be taken by the aeroplane.

Another way to decrease the temperature is by changing the combustion process itself. This can be achieved by changing the shape of combustion chambers. The highest temperatures in a combustion chamber occur approximately at stoichiometric fuel-air-ratio. (When the specific heat at constant pressure of either the fuel or the oxidant is lower, the highest temperatures will occur slightly on the fuel-rich side or fuel-lean side.) Thus, the NO_x-emissions can be reduced by avoiding combustion at stoichiometric conditions. Burning in fuel-rich conditions can also reduce NO_x emissions because of the low O radical concentrations.

Especially diffusion flames have a nearly stoichiometric flame zone. When liquid fuels are used, stoichiometric conditions often occur around the evaporating droplets. Therefore, NO_x-reductions can be achieved by vaporising liquid fuels before they are burnt. Another way to avoid stoichiometric conditions is by thorough mixing of the fuel and oxidant prior to the combustion process: in this so-called lean-premixed (prevaporised) combustion chamber, the combustion equivalence ratio is (substantially) lower than one. The last way to avoid stoichiometric zones mentioned here is by application of a so-called rich-quench-lean combustion chamber. In the first part of this combustion chamber, fuel-rich burning occurs. Because of the relatively low temperatures and low oxygen concentrations, NO_x formation remains low. After this, a lot of oxidant is mixed with the products of the fuel-rich combustion causing the flame to quench. In the last part of the combustion chamber, the fuel-rich products

and the oxidant that is left, are lighted and burn in a fuel-lean mixture. In figure F.1, F.2 and F.3, a conventional combustion chamber, using a diffusion flame as well as a lean-premixed and a rich-quench-lean combustion chamber are shown as well as possible multi-reactor models for these combustors.

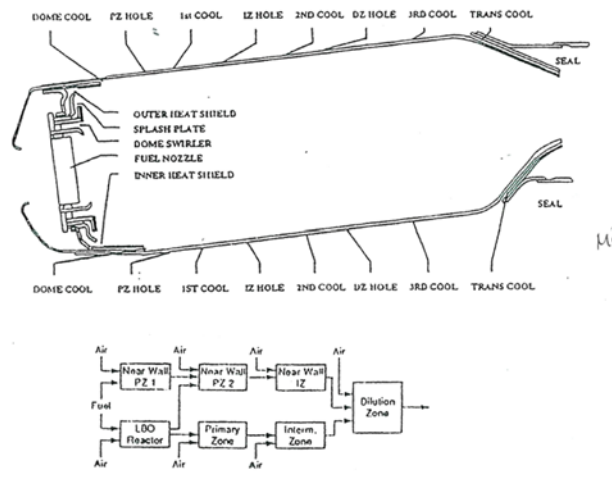


Figure F.1 Picture and flow model of an annular diffusion flame combustor

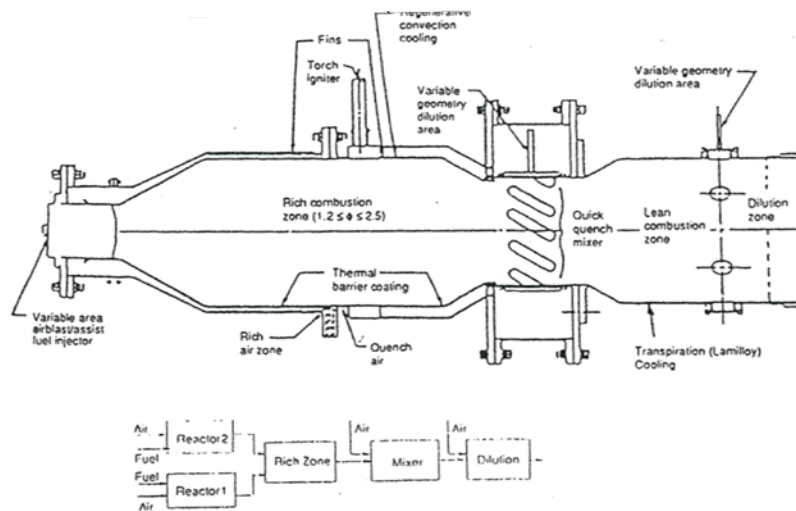


Figure F.2 Picture and flow model of a rich-quench-lean combustor

Appendix G An introductory description of GSP

In this appendix, some introductory remarks on GSP are made. For more information on GSP, one should read (Visser, Ref. 43).

G.1 A general introduction to GSP

The development of the Gas turbine Simulation Program (GSP) started in 1986 by W. Boumans at the Department of Aerospace Engineering at Delft University of Technology (DUT). The base for GSP was NASA's Dyngen, which was used before GSP at DUT, but showed deficiencies. Improvements in GSP were made at the National Aerospace Laboratory (NLR), where GSP was converted from ANSI FORTRAN-5 to DELPHI (object oriented Pascal, Windows 95/NT application).

GSP is a so-called 'component stacking program': gas turbine configurations can be built by putting together predefined components. These components include primary gas turbine components, like a compressor, a turbine and a heat exchanger, but also control units, like a power lever and a unit used to control the area of the exhaust of afterburning engines. The components are primarily modelled in a zero-dimensional way, which means that only conditions at the inlet and outlet of the components are calculated. As an example a GSP model of an afterburning engine is shown in figure G.1.

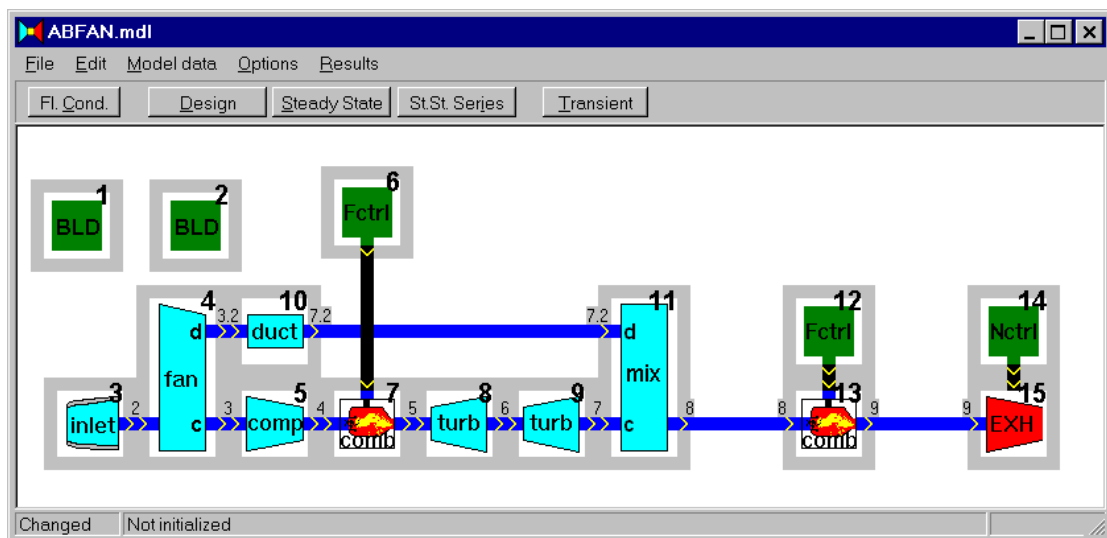


Figure G.1 A GSP gas turbine model

Once the components have been selected, various input design data are required, like pressure ratios, rotor speeds, etc. After the insertion of data, a design point calculation can be performed. In the design point calculation, the design thermodynamic cycle and also the relevant geometry

of the gas turbine is calculated. GSP is primarily used as an *off-design* gas turbine performance simulation program. This means that for a given gas turbine, a model is built using (measured) design point data, and performance prediction is calculated for off-design operating points. The program can handle two sorts of off-design conditions: steady state series (or points) and transients.

In steady state series a number of working points is calculated, assuming stationary working conditions of the gas turbine. Therefore, all the acceleration terms in equations of motion are made zero, so moments of inertia and mass do not matter in these calculations. Because in a stationary working point the mass flow entering a component is equal to the mass flow leaving it, volumes are not important either. Steady state calculations can easily be used to find the relation between different gas turbine quantities in part-load situations. To find this, the easiest way is to decrease the fuel flow gradually from the maximum value to the minimum value. The program then calculates a range of working points. In every working point a large number of quantities can be calculated and selected as output variables. After the calculations GSP offers the possibility of making graphs with varying data on both axes.

In a transient simulation, a gradual change of situations is calculated, involving accelerations and temporary accumulations of gases (eventually with condensed species) in components. In this case, mass, volumes and moments of inertia do matter. Time only matters in transients, not in steady state series.

G.2 GSP and the gas model

During all cycle calculations, the program goes through the gas turbine by solving equations from the inlet to the exhaust. Each component involves a number of equations to be solved. The inlet data for each component (except for the inlet and the fuel inlet of the combustion chamber), are the outlet data of the component directly before the component considered. Thus, data are passed on from component to component, describing the properties of the medium used in the gas turbine. In GSP 7.0, among others, the following quantities are passed on from component to component:

- mass flow,
- fuel-air-ratio,
- total temperature,
- total pressure.

In a design calculation, the design cycle is calculated, as well as the relevant flow areas in the gas turbine. As a result, after the design point calculation, the geometry of the engine is fixed. A

clear distinction has to be made between *design* data and *off-design* data. Design data determine the design cycle and the geometry of the gas turbine, while off-design data only determine separate working points. If design data are changed and a new design calculation is performed, a *different* gas turbine 'is designed'.

Once the gas turbine geometry is fixed, steady state series and transients can be performed. During steady state series and transients, GSP uses a certain number of state variables and error variables. The number of both is fixed per component, though it is not the same for all components. State variables are the values of (primarily thermodynamic) gas turbine quantities relative to the value of these quantities under design conditions. The number of state variables is exactly the number of variables needed to solve the equations adhering to each component and thereby to calculate the outlet conditions from the inlet conditions. Examples of state variables are rotor speeds and the turbine pressure ratio relative to their values under design conditions. The value of the state variables and thereby the working conditions of the gas turbine are found, using the error variables, as follows. An initial guess of the state variables is made. Then GSP calculates a cycle, using user specified quantities, and a number of deviations from the state variables are found. These deviations determine the value of the error variables. Using the values of the error variables new values for the state variables are calculated, and another cycle is calculated which results in other values of the error variables. In this way, corrections to the state variables are made until the error variables are considered small enough. Then the state variables are known and so is the working point of the gas turbine. All other desired quantities that were not necessary to find the working point are calculated now. If a number of subsequent points are calculated, the state variables at the last point calculated before the one to be calculated can be used as initial guesses for the state variables.

Appendix H Details of the new GSP 8.0 gas model

In this appendix a detailed description of the GSP 8.0 gas model is given. For more details, the Pascal code (with comments) in appendix K can be consulted.

H.1 General remarks

To calculate the relevant thermodynamic properties of the medium used by the gas turbine, NASA polynomials are used. The NASA coefficients a_i up to and including a_8 , b_1 and b_2 , determining the specific heat at constant pressure, enthalpy and steady state entropy as a function of temperature, are tabulated for all species, as well as coefficients used to find the viscosity (see appendix C). Finally, the molar mass and the heat of formation are also added. For water, the polynomials are only valid until 600 (K). The critical temperature for water is 647.29 (K). In figure H.1, the c_p of water is given as a function of temperature. Although the rise in c_p between 600 (K) and 647.29 (K) is quite steep (may be too steep), the polynomial is assumed to be valid until 647.29 (K), because the amounts of water present above 600 (K) will be very small. Because the number of coefficients should be the same for all species, including liquid water, coefficients for the temperature range above 1000 (K) have to be inserted into the program. The coefficients for the polynomial valid above 1000 (K) are the same as for the polynomial valid below 1000 (K) (actually to 600 (K) of course). Because at temperatures higher than the critical temperature no water will be present, these coefficients are never used and act as dummy-coefficients.

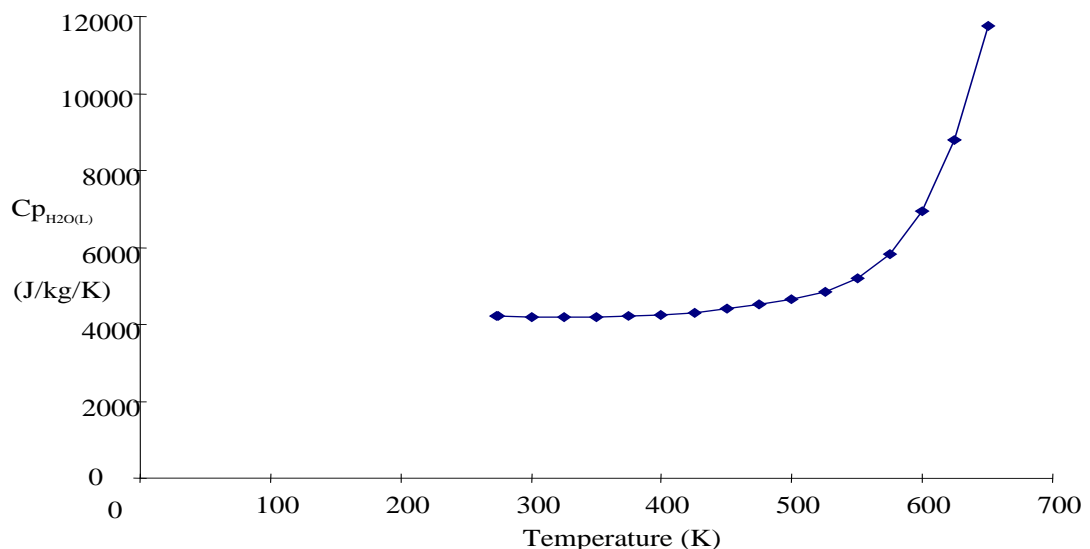


Figure H.1 Specific heat at constant temperature (c_p) of water as a function of temperature

H.2 Determination of thermodynamic and thermal transport properties as a function of temperature, pressure and composition

Like in CEA, the perfect gas law is assumed to be valid even when small amounts of condensed species (up to several percent by weight) are present. Because condensed species are assumed to occupy a negligible volume relative to the gaseous species, volumes refer to gases only, while mass includes condensed species.

The specific heat at constant pressure

The specific heat at constant pressure at a given temperature is first calculated separately for each of the species present in the mixture, using the polynomial (C.22). The specific heat at constant pressure for the mixture is then found after applying (C.23).

The enthalpy

The enthalpy is calculated in the same way as NASA. First the enthalpy per specie is calculated using equation (C.25), and subsequently the enthalpy for the mixture is found using equation (C.26).

The specific gas constant

The specific gas constant of a mixture is calculated by multiplying the number of moles of gaseous species per gram of mixture (n), calculated using equation (C.28), and the universal gas constant, R , like in the CEA-program:

$$R_g = nR \quad (\text{H.1}).$$

If the mixture only contains gases, the value of the above equation is the same as the value of the universal gas constant divided by the molar mass of the mixture (in this case both molar masses used by NASA, calculated with equations (C.29) and (C.30) are the same). For mixtures containing condensed species the specific gas constant is the same as the universal gas constant divided by the molar mass defined by equation (C.29). By using the above equation (H.1), the well-known expression for the speed of sound automatically becomes the same as equation (C.39).

The entropy

The entropy is again calculated using the same formulas as the CEA-program uses. The incoming mass fractions are first converted into mole fractions. After that the total mole fraction of the gaseous species is calculated (needed for equation (C.32)). The next step is to calculate the steady state standard entropy per specie divided by the universal gas constant S_j^0/R , using

equation (C.34). With the pressure known, the entropy divided by the universal gas constant can be calculated for each of the gaseous species using equation (C.32). Then, the entropy contribution of the gaseous part of the mixture divided by the universal gas constant can be found by taking the weighted average of the entropy values per specie using the mole fractions. For water, the standard state entropy also follows from equation (C.34), using the universal gas constant. According to equation (C.33) the entropy per specie for condensed species is equal to their standard state entropy. The contribution of the (liquid) water is then added to the entropy (divided by the universal gas constant) of the gaseous part. Multiplication with the universal gas constant and division by the molar mass (g/mole) are needed to find the entropy for the mixture.

The ratio of specific heats

The value of γ is calculated in the same way as in GSP 7.0, using c_p for frozen composition and the specific gas constant calculated as defined above by equation (H.1).

The speed of sound

The speed of sound is calculated in the same way as in GSP 7.0. This is practically the same as in the NASA CEA-program, although the ratio of specific heats (γ) is used instead of (γ_s). The reason for this is that the ideal gas law is assumed to be valid, resulting in equal values for γ and γ_s (see equations (C.37) and (C.38)). Because of equation (H.1), ' nR ' in equation (C.39) is equal to R_g . Therefore, equation (C.39) is found to be equal to (C.11).

The dynamic viscosity

The only thermal transport property calculated in the new gas model is the dynamic viscosity. It is calculated in almost the same way as done in the NASA program. The mass fractions have to be converted to mole fractions first. Once this is done, liquid species (water) are omitted and the other mole fractions are increased so that the sum of the gaseous mole fractions equals one. Then for all the non-zero fraction species, equation (C.40) is used to find the dynamic viscosity per specie. Subsequently, for each component the viscosity interaction coefficient with the other components is calculated using equation (C.43), and equation (C.41) is used to find the dynamic viscosity of the mixture. Equation (C.42) is not used, because the interaction parameters needed are given in CEA only for a number of components. Therefore, extra data for the available interaction parameters would have to be inserted in GSP, and switches and new procedures would become necessary to check whether interaction parameter data would be available and to calculate them. Because the gain of applying these extra switches and procedures is assumed to be small, the interaction parameters are not used.

H.3 Modelling of composition changes

Like the NASA gas model, the GSP 8.0 gas model has a limited description of kinetics, only using frozen or equilibrium compositions. If the processes are assumed to be slow in comparison with the residence times in the components, a frozen composition is assumed. Otherwise, the composition will change and the equilibrium composition is calculated.

In gas turbines, the composition of the working medium can change due to changing temperatures and pressures if evaporation (/condensation) or dissociation occurs. Because evaporation and dissociation involve conversion between potential energy and heat, evaporation and dissociation also influence temperature itself. In order to find the composition and temperature when evaporation or/and dissociation occur, the assumption is made that there is no heat loss (i.e. adiabatic reactions) or pressure loss during the evaporation and dissociation. The temperature and composition are then found in an iteration procedure solving the equilibrium composition belonging to a certain temperature and checking the enthalpy balance (also called energy equation) to see whether the temperature is correct.

H.3.1 Solving the enthalpy balance

The temperature is found assuming that the enthalpy before and after the process is equal (i.e. the process is adiabatic). If the compositions of mixtures change, there are not only enthalpy changes due to temperature changes, but there is also an enthalpy change due to the chemical reaction (see appendix B).

To find the temperature after a change in composition, this enthalpy change due to reaction must be taken into account. In the proposed gas model, the following procedure is applied, shown schematically in figure H.2.

In this figure, the left side denotes the temperature and composition before the reaction takes place and the right side after the reaction has taken place. In the vertical direction, temperature varies. The arrows denote enthalpy changes because of processes (reactions or heating or cooling of fixed compositions).

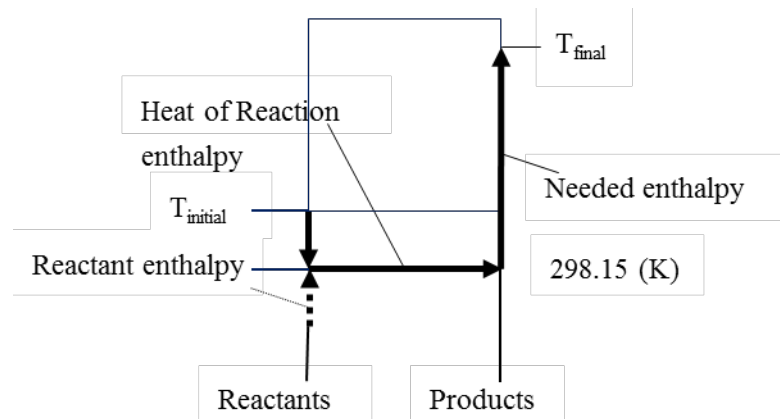


Figure H.2 The relevant enthalpy changes used to solve the enthalpy balance (Kuo, Ref. 26)

First, the composition of the products after the process has taken place has to be known. If the composition is a function of temperature, a guess of the composition can be used. Once this composition is found three enthalpies are calculated. The first one is the enthalpy that is obtained when the reactants are brought from their initial temperature T_{initial} (of course, separate reactants can also have different temperatures) to a temperature of 298.15 (K). The initial temperature can also be lower than 298.15 (K) (see T_{initial}). This enthalpy is called the reactant enthalpy. At 298.15 (K) the reaction (change of composition) is assumed to take place. From the expected products of the reaction and the reactants, the heat of reaction can be calculated using the heats of formation of reactants and products (see appendix B for an example). The sum of these two enthalpies is the enthalpy available to warm the products to the new temperature. So the correct temperature after the reaction is found if the available enthalpy is equal to the enthalpy needed to bring the mixture to a new temperature T_{final} .

In the above description, three separate enthalpies are calculated to check the enthalpy balance. However, because the (absolute value of the) enthalpy is defined in the same way as in the NASA CEA-program, the assumption of adiabatic reactions simply results in equal summation of enthalpies before and after the reactions.

This can be shown as follows. In the NASA CEA-program, the absolute value of the enthalpy is defined by the fact that at 298.15 (K), the enthalpy of a specie must be equal to its heat of formation at 298.15 (K). Therefore, the enthalpy for a specie can be written as:

$$h(T) = H_{f,298.15} + \int_{T'=298.15}^T c_p(T') dT' \quad (\text{H.2}),$$

where: $H_{f,298.15}$ = heat of formation at 298.15 (K).

This results in the following expression for the enthalpy change involved in a temperature change from T to 298.15 (K) (for one specie):

$$\begin{aligned}
 h(T) - h(298.15) &= H_{f,298.15} + \int_{T'=298.15}^T c_p(T') dT' - \left(H_{f,298.15} + \int_{T'=298.15}^{298.15} c_p(T') dT' \right) \\
 &= \int_{T'=298.15}^T c_p(T') dT'
 \end{aligned} \quad (H.3).$$

Next, a reaction where reactants A and B, at temperatures of respectively T_A and T_B undergo a reaction and form products C and D, both at temperature T_{final} , is considered. Equating the sum of reactant enthalpy and the heat of reaction to the needed enthalpy gives:

$$\begin{aligned}
 \int_{T=298.15}^{T_A} c_{p_A}(T) dT + \int_{T=298.15}^{T_B} c_{p_B}(T) dT + \left\{ (H_{f,298.15_A} + H_{f,298.15_B}) - (H_{f,298.15_C} + H_{f,298.15_D}) \right\} = \\
 \int_{T=298.15}^{T_{final}} (c_{p_C}(T) + c_{p_D}(T)) dT
 \end{aligned} \quad (H.4).$$

Rearranging the terms in this equation gives:

$$\begin{aligned}
 H_{f,298.15_A} + \int_{T=298.15}^{T_A} c_{p_A}(T) dT + H_{f,298.15_B} + \int_{T=298.15}^{T_B} c_{p_B}(T) dT = \\
 H_{f,298.15_C} + \int_{T=298.15}^{T_{final}} c_{p_C}(T) dT + H_{f,298.15_D} + \int_{T=298.15}^{T_{final}} c_{p_D}(T) dT
 \end{aligned} \quad (H.5).$$

In other words (using (H.2)):

$$h_A(T_A) + h_B(T_B) = h_C(T_{final}) + h_D(T_{final}) \quad (H.6).$$

Studying the last equation, it is obvious that the enthalpies (defined according to equation (H.2)) before and after the reaction must be equal for adiabatic reactions.

Although it is easier to solve the enthalpy balance just by calculating the enthalpy values than by separately calculating three enthalpies, this is not done in GSP 8.0. The main reason for this is that the method calculating the three enthalpies is more general. In particular, it is independent of the absolute values of the enthalpy. Therefore, less adaptations have to be made to the program if users want another definition for the absolute value of the enthalpy (e.g. the enthalpy can be made zero at a temperature of 200 (K)).

H.3.2 Determination of equilibrium compositions at a given temperature

Evaporation/condensation

The calculation of the equilibrium state is as follows. A polynomial is used to predict the maximum vapour pressure of water. This polynomial is taken from (Ruijgrok, Ref. 39). It is approximately valid until the critical temperature of water (647.29 (K)).

First, the temperature is compared to the critical temperature of water. If the temperature is higher than the critical temperature, all the water in the mixture is vaporised. If the temperature is lower than the critical temperature, the maximum vapour pressure is compared with the vapour pressure of water if all the water were gaseous. If the maximum vapour pressure is higher than this vapour pressure, all the water will be vaporised. If the maximum vapour pressure of water is exceeded, a part of the water must become liquid.

Dissociation

As stated in appendix B (paragraph 4), an equilibrium can be calculated either by optimisation of a thermodynamic function (e.g. entropy or free energy) or by using equilibrium constants (see for example Gordon, Ref. 15). The last method is applied here, because the number of equations to be solved is not very big.

In chapter 4, it was stated that in the gas model, (outside the combustion chamber) only dissociation of water and carbon dioxide to hydrogen, carbon monoxide and oxygen were considered, according to equations (4.1) and (4.2). These two reactions lead to the following equations for the equilibrium constants to be solved (see appendix B, paragraph B.4):

$$K_{p,CO_2} = \frac{n_{CO} \sqrt{n_{O_2}}}{n_{CO_2}} \sqrt{\frac{p}{N}} \quad (H.7),$$

$$K_{p,H_2O} = \frac{n_{H_2} \sqrt{n_{O_2}}}{n_{H_2O}} \sqrt{\frac{p}{N}} \quad (H.8).$$

Because there are five unknown concentrations, three additional equations are needed to determine the composition. These three equations are the atom conservation equations for C, H and O:

$$(n_{CO_2} + n_{CO})_{before} = (n_{CO_2} + n_{CO})_{after} \quad (H.9),$$

$$(n_{H_2O} + n_{H_2})_{before} = (n_{H_2O} + n_{H_2})_{after} \quad (H.10),$$

$$(2n_{O_2} + 2n_{CO_2} + n_{CO} + n_{H_2O})_{before} = (2n_{O_2} + 2n_{CO_2} + n_{CO} + n_{H_2O})_{after} \quad (H.11).$$

In these equations ‘before’ and ‘after’ mean before reaction and after reaction. Conservation of N is automatically taken care of, because the amounts of nitrogen containing species (N_2 , NO and N_2O) are assumed to be constant. This approach is only valid until temperatures of around 2200 (K). Whether NO is really constant is the question. NO formation and depletion is thoroughly discussed in chapter 6, where the emission model is shown.

For temperatures above 2200 (K), another (similar) procedure can be used, which is a part of the combustor model, and is to be described in appendix J. In this model, three additional species may be present: O, H and OH. Also, *equilibrium* concentrations of NO and N_2O are calculated, but these are only needed for the emission model, which is part of the combustor model. Because the temperatures outside of the combustion chamber will not likely exceed 2200 (K), this last procedure is only used in the combustion chamber.

In both the mentioned procedure for dissociation and the procedure which is part of the combustor model, a number of equations (equilibrium constants and conservation of species equations) need to be solved to find a number of fractions. In both procedures, this is done in the same way. First, an initial guess is made of a fraction. Because the number of unknown fractions is equal to the number of equations, the other unknown fractions can be calculated for this guess and also a new value is found for the guessed fraction. The correct fraction is found if the new value for the fraction is equal to the guessed one. Therefore, the deviation of the new value from the guessed one is used to find the correct fraction and therefore the correct equilibrium composition.

The choice of the fraction to be guessed is determined by the fact that it must be a non-zero (because of limited Delphi accuracy not below $1e-15$) fraction under all circumstances, otherwise, the equations can't be solved. Species containing C and H-atoms can't be chosen, because it must be possible to burn hydrogen in air (not containing carbon dioxide) and to burn fuels only containing carbon (e.g. pure carbon monoxide). In those cases, the guessed fraction would be zero. Taking nitrogen or argon leads to problems if not air is used, but pure oxygen. Therefore, the most logical fraction is the O_2 -fraction, because it will always be present in gas turbine cycles comprising combustion processes. Another reason to choose the O_2 -fraction is because it couples the equations for the equilibrium constants ((H.7) and (H.8)). If the O_2 -fraction is guessed, both equations can be solved separately.

However, problems will be encountered when the O₂-fraction becomes smaller than 1e-15, because of the limited (standard) accuracy of Delphi. This problem could be solved by increasing the accuracy of the Delphi-calculations. However, this will take more calculation time, and O₂-fractions lower than 1e-15 usually occur only for (very) rich mixtures (equivalence ratio above (about) 1.7) especially at low temperatures, which are not likely to occur in gas turbines, especially not outside the combustion chamber.

H.4 Total versus static temperatures and pressures

In gas turbine performance calculations, total temperatures and pressures are often used. The relations between total and static temperature and total and static pressure are:

$$T_t = T_s \left\{ 1 + \frac{\gamma - 1}{2} M^2 \right\} \quad (\text{H.12}),$$

$$P_t = P_s \left\{ 1 + \frac{\gamma - 1}{2} M^2 \right\}^{\frac{\gamma}{\gamma - 1}} \quad (\text{H.13}).$$

Table H.1 Ratios of total and static temperature and pressure for various gammas

Gamma:	1,01		1,1		1,2		1,4		1,6	
Mach:	Tt/Ts	Pt/Ps	Tt/Ts	Pt/Ps	Tt/Ts	Pt/Ps	Tt/Ts	Pt/Ps	Tt/Ts	Pt/Ps
0,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
0,10	1,00	1,01	1,00	1,01	1,00	1,01	1,00	1,01	1,00	1,01
0,20	1,00	1,02	1,00	1,02	1,00	1,02	1,01	1,03	1,01	1,03
0,30	1,00	1,05	1,00	1,05	1,01	1,06	1,02	1,06	1,03	1,07
0,40	1,00	1,08	1,01	1,09	1,02	1,10	1,03	1,12	1,05	1,13
0,50	1,00	1,13	1,01	1,15	1,03	1,16	1,05	1,19	1,08	1,21
0,75	1,00	1,33	1,03	1,36	1,06	1,39	1,11	1,45	1,17	1,52
1,00	1,01	1,65	1,05	1,71	1,10	1,77	1,20	1,89	1,30	2,01

In these equations, M is the Mach number and the subscripts 't' and 's' denote total and static respectively. In table H.1 the differences between static and total temperatures and pressures are calculated for a number of γ 's and Mach numbers.

The static temperature of gases is due to Brown's heat movement and can be seen as a measure of energy contained in the molecules because of their speed in random direction (a vibration around the equilibrium composition). The difference between total and static temperatures is due to the speed that all the molecules have in common. Therefore, the total temperature of the gases is a sort of (static) temperature taking into account both aforementioned speeds. Because

of this, total temperatures are often used to calculate thermodynamic properties as if they were static temperatures. Because the Mach numbers and γ 's in gas turbines aren't very big, the difference between total and static temperatures remains limited. A second reason to use total temperatures is that they remain constant in adiabatic processes (no heat loss), no matter what happens with the flow velocity. A number of processes in the gas turbine (like compression in the inlet) can be assumed adiabatic with reasonable accuracy.

For total pressures to remain constant however, processes need to be isentropic, which can hardly be assumed to be the case in gas turbine components. The differences between total and static pressures are bigger than the differences between total and static temperatures, as can be seen from the power in equation (H.13) ($\gamma > 1$) and table H.1. However, total pressures are often used in gas turbine performance calculations.

In GSP 7.0, almost everywhere total temperatures and pressures are used to describe the gas conditions. Only if geometric data of the engine are available, because the speed of the gases is needed, like in the exhaust, static data are used.

In GSP 8.0, a few extra procedures are used compared to GSP 7.0. The procedures used to calculate composition changes due to evaporation and dissociation need pressures and temperatures. For the same reasons as described above, total temperatures are used here as if they were static temperatures. An important issue however, is whether the pressures used by the procedures have to be static or total. To calculate static pressures from total conditions, geometric data of the gas turbine (e.g. areas) have to be known. Because geometric data are gas turbine specific, it is preferable not to use these data, because it means that more data have to be specified.

Therefore, in GSP 8.0 total pressures and temperatures are used at locations where no geometric data are known, i.e. everywhere except in the combustion chamber and the exhaust. The modelling of the combustion chamber is described in chapter 6.

Appendix I Calculation of compressor and turbine performance with real gas effects

I.1 Calculating outlet conditions

In the compressor and turbine, the isentropic efficiency is used to calculate the conditions at the outlet. If the composition is constant, this is done as indicated in figure I.1. The f denotes a functional relationship.

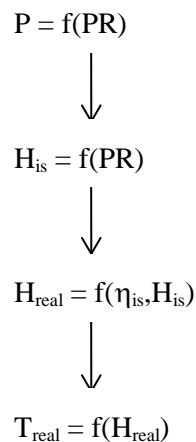


Figure I.1 Calculation of the outlet conditions assuming constant composition

The pressure ratio is either user-specified in the case of the design point or it is treated as a state variable (see paragraph G.2). The outlet pressure is found by multiplying the inlet pressure by the pressure ratio. Using the pressure ratio and the inlet conditions, the isentropic enthalpy change can be found and multiplying this with the isentropic efficiency yields the real enthalpy change. From this enthalpy change, the temperature at the outlet can be calculated.

Because the composition is variable now, the first interesting question is whether the varying composition is accounted for in the isentropic efficiency or not. In other words, is the isentropic efficiency defined using a constant composition or a changing composition? This is an important issue because a changing composition yields a changing entropy. It is to be expected that the isentropic efficiency is determined under conditions where the composition remains unchanged. Possibly, the changing compositions could lead to (isentropic) enthalpy changes too much different from (isentropic) enthalpy changes in case of constant composition, because the changing composition can have a big effect on entropy changes. Therefore, the isentropic efficiency is used assuming that it is defined for constant composition.

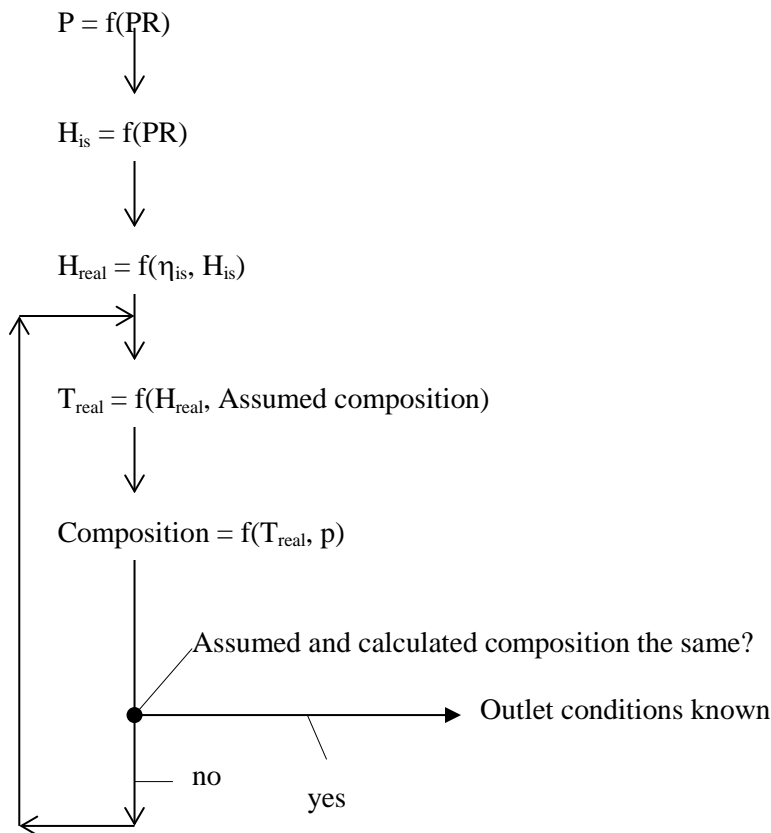


Figure I.2 Calculation of the outlet conditions assuming variable composition

The outlet conditions of the compressor and turbine are calculated as follows, see figure I.2. As described above, using the pressure ratio, the inlet conditions and the isentropic efficiency, the enthalpy at the outlet is found. A check is made to see whether the assumed (constant) composition can exist at outlet conditions (temperature and pressure). If this is the case, the outlet conditions are known. If not, the newly calculated composition is used to find a new value for the temperature. The composition at this temperature is determined and compared with the composition found at the last iteration (the assumed composition). This process is repeated until the difference between the two compositions is considered small enough.

I.2 Using maps

Because the compositions have become variable, adaptations are needed when using maps to predict off-design performance of compressors and turbines. A first important question is whether maps can also be used when liquid species (e.g. water) are present. The assumption is made that they are valid, because otherwise (without maps), off-design behaviour could not be described at all. Once the maps are assumed to be valid, the question rises how the dimensionless parameters in the map should be calculated when different (two-phase)

compositions are encountered. The relevant dimensionless parameters are the component (isentropic) efficiency (η), the dimensionless number of rotations, the dimensionless mass flow and finally the pressure ratio. To account for the effects of boundary layers on component performance, the isentropic efficiency can be corrected for changing Reynolds numbers, because the dynamic viscosity is calculated in the new GSP 8.0 gas model.

The definition of the component (isentropic) efficiency for varying compositions has been described above. The pressure ratio only depends on pressures and therefore remains unchanged if (only) compositions change. To find the way to account for changing compositions in the dimensionless number of rotations and the dimensionless mass flow, the reason why maps, and the dimensionless parameters appearing in maps, are used should be clarified first (Cohen, Ref. 9).

The non-dimensional method of plotting characteristics is based on the assumption that components will yield the same performance (in terms of pressure ratios, temperature ratios and isentropic efficiencies) if similar velocity triangles are encountered in the compressor or turbine. To achieve these similar velocity triangles, two Mach numbers should be the same, the Mach number of the rotor blade tips and the flow Mach number.

The dimensionless number of rotations

The dimensionless number of rotations represents the Mach number of the rotor blade tips. The Mach number of the rotor blade tips can be calculated using:

$$M_R = \frac{NL}{\sqrt{\gamma R_g T}} \quad (I.1),$$

where: M_R = Mach number of rotor blade tips (-),
 N = number of rotations in time (rad/s),
 L = characteristic length (m).

It is obvious that the numerator of this expression is the rotor tip speed and the denominator is the speed of sound. For the characteristic length, a diameter in the engine is usually used. Because this diameter is constant, it is left out of the dimensionless number of rotations, which is therefore not truly dimensionless. If the composition in the component is constant, the values for γ and R_g are the same for all working points and these can be omitted from the above expression too. If the composition can change, however, γ and R_g are also variables. Therefore, the following expression should stay the same to retain the same Mach number of the rotor blade tips:

$$\frac{N}{\sqrt{\gamma R_g T}} \quad (I.2).$$

Because GSP doesn't use dimensionless parameters, but corrected numbers, the following expression will be used for maps in the new version of GSP:

$$N_c = \frac{N}{\sqrt{\frac{\gamma R_g T}{\gamma_{st} R_{g,st} T_{st}}}} \quad (I.3).$$

In this expression, the subscript 'st' stands for standard conditions.

The dimensionless mass flow

The dimensionless mass flow represents the flow Mach number. Using

$$m = \rho V A \quad (I.4),$$

where:

m	= mass flow (kg/s),
ρ	= density (kg/m ³),
V	= flow velocity (m/s),
A	= flow area (m ²),

and the ideal gas law, the following expression can be derived for the flow Mach number:

$$M_F = \frac{V}{a} = \frac{m \sqrt{T}}{A p} \sqrt{\frac{R_g}{\gamma}} \quad (I.5).$$

The area A, taken somewhere in the gas turbine is usually constant, so it can be omitted. Again, if the composition can change within the component considered, γ and R_g are not constant and therefore cannot be omitted. So for the corrected mass flow the following formula can be used:

$$m_c = \frac{m \sqrt{\frac{T}{T_{st}}}}{\frac{p}{p_{st}}} \sqrt{\frac{R_{g,st} \gamma_{st}}{R_{g,st} \gamma}} \quad (I.6).$$

The Reynolds number

A new feature in GSP 8.0 is that the influence of different Reynolds numbers on isentropic efficiency can be modelled. The Reynolds number is defined as:

$$Re_L = \frac{\rho V L}{\mu} \quad (I.7),$$

where:

ρ	= density (kg/m ³),
V	= flow speed (m/s),
L	= characteristic (geometric) length (m),
μ	= dynamic viscosity (kg/m/s).

This quantity indicates the relative importance of the shear and the inertia forces within the flow; the lower the value of the Reynolds numbers, the more important (relatively) are the viscous forces. As a result, lower Reynolds numbers will be accompanied by bigger boundary layers. These will have a negative influence on the component efficiency.

To denote the influence of Reynolds number on efficiency, the dimensionless Reynolds Number Index (RNI) is used. This is the actual Reynolds number divided by the Reynolds number of air under standard conditions. Because the characteristic length is usually a constant length somewhere in the gas turbine, it cancels out. The effect of speed on component performance is already accounted for in the dimensionless mass flow and dimensionless number of rotations. Therefore, the effect of speed on Reynolds Number Index is not taken into account. The following equation is found for the Reynolds number index:

$$RNI = \frac{Re_L}{Re_{L,st}} = \frac{\frac{\rho}{\mu}}{\frac{\rho_{st}}{\mu_{st}}} \quad (I.8).$$

Using the perfect gas law, this relation becomes:

$$RNI = \frac{\frac{p}{T} \frac{R_g}{R_{g,st}} \frac{\mu}{\mu_{st}}}{\frac{p_{st}}{T_{st}} \frac{R_g}{R_{g,st}} \frac{\mu}{\mu_{st}}} \quad (I.9).$$

All quantities in this equation are calculated in the gas model.

Appendix J Calculating combustion equilibrium

The way of calculating the equilibrium composition and temperature is the same as used in the gas model, described in paragraph H.3. For a given temperature, the equilibrium composition can be calculated. To see if this temperature is correct, the enthalpy balance is solved.

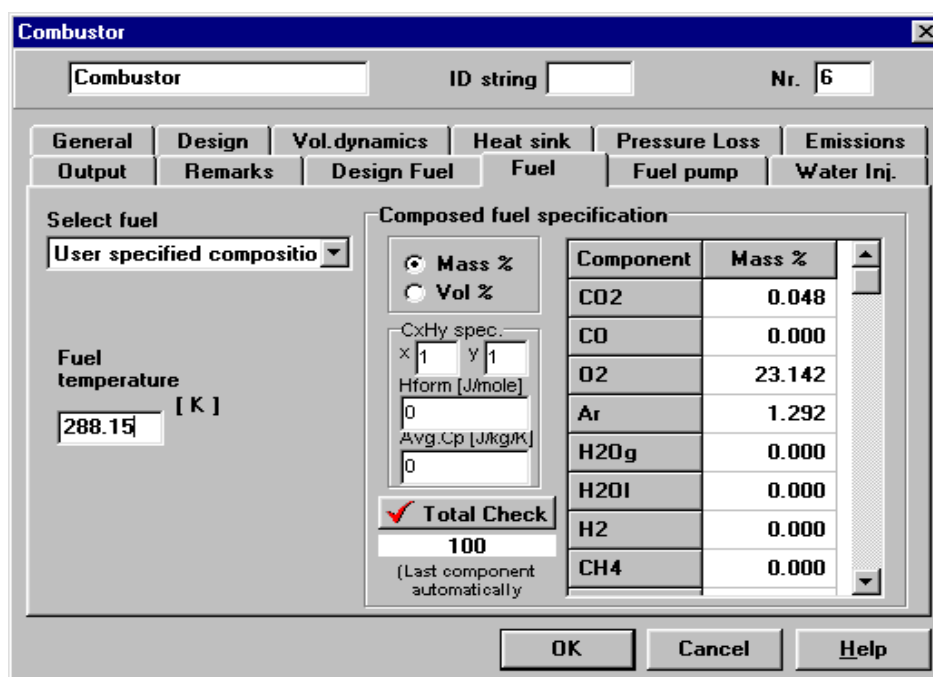
J.1 Solving the enthalpy balance

In this paragraph, it is described how to calculate the relevant enthalpies, shown in figure H.2.

In GSP 8.0 two types of fuels can be specified:

- fuels with complete composition specified,
- fuels with composition characterised by H/C-ratio.

Fuels with complete composition specified



The screenshot shows the 'Combustor' software window with the 'Fuel' tab selected. The 'Select fuel' dropdown is set to 'User specified composition'. The 'Fuel temperature' is entered as 288.15 K. The 'Composed fuel specification' section shows 'Mass %' selected, with a 'Total Check' of 100. A table lists the components and their mass percentages:

Component	Mass %
CO2	0.048
CO	0.000
O2	23.142
Ar	1.292
H2Og	0.000
H2Ol	0.000
H2	0.000
CH4	0.000

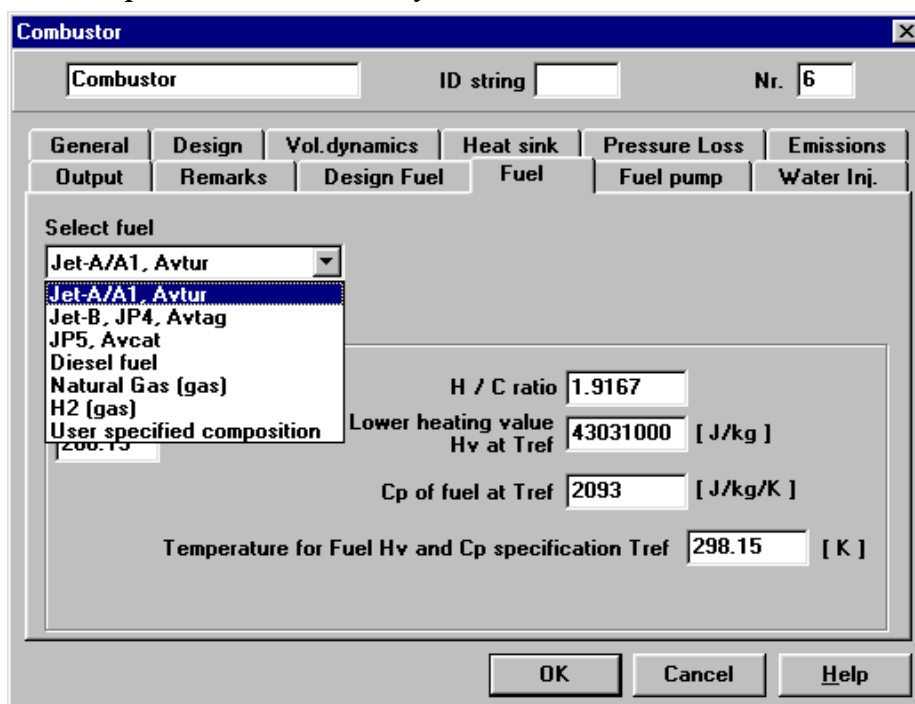
Figure J.1 User interface for fuel with user specified composition

To start with, the available enthalpy has to be calculated as the sum of the reactant enthalpy and the heat of reaction. For these fuels, the composition can be specified in terms of (mass or volume) fractions of the following species: (CO₂, CO, O₂, Ar, H₂O(g), H₂O(l), H₂, CH₄, C₂H₆, C₂H₄, C₃H₈, C₄H₁₀, O, H, OH, C_xH_y, O₂). The user interface for these fuels is (partly) shown in figure J.1. The component C_xH_y allows users to specify a hydrocarbon component different from the specified ones. For all these species, except for C_xH_y, thermodynamic data are present

in GSP (see chapter 4 and appendix H). Therefore, for C_xH_y some extra data need to be specified by the user: the values of 'x' and 'y', the heat of formation at 298.15 (K) and the average c_p between the fuel injection temperature and 298.15 (K). The reactant enthalpy can easily be calculated using the user specified fuel temperature and formulae (C.25) and (C.26) to find the enthalpy for the composition (except for C_xH_y) at both the temperatures. The enthalpy change for C_xH_y is calculated by multiplying the c_p from the interface by the temperature difference between fuel injection temperature and 298.15 (K).

If the product composition is known, the heat of reaction is found as the difference between the summation of formation enthalpies of the reactants and products.

Fuels with composition characterised by H/C-ratio



Combustor

Combustor ID string Nr. 6

General Design Vol.dynamics Heat sink Pressure Loss Emissions
Output Remarks Design Fuel Fuel Fuel pump Water Inj.

Select fuel

- Jet-A/A1, Avtur
- Jet-B, JP4, Avtag
- JP5, Avcad
- Diesel fuel
- Natural Gas (gas)
- H2 (gas)
- User specified composition

H / C ratio 1.9167

Lower heating value Hv at Tref 43031000 [J/kg]

Cp of fuel at Tref 2093 [J/kg/K]

Temperature for Fuel Hv and Cp specification Tref 298.15 [K]

OK Cancel Help

Figure J.2 User interface for standard fuel (composition not exactly known)

Apart from fuels with a specified composition, a number of other fuels can be chosen: jet fuels (Jet-A, Jet-B (=JP-4) and JP-5), diesel, hydrogen and natural gas. The user interface for these fuels is shown in figure J.2. (The fuel temperature is not completely visible due to the pull-down menu on the left.) The composition of jet fuels, diesel and natural gas, which are assumed to consist completely of hydrocarbons, is characterised by (molar) H/C ratio, to be specified by the user. For calculation reasons, the chemical formula for these fuels is $C_{(1)}H_{H/C}$. (This approach is also used for jet fuels in the NASA CEA-program (McBride, Ref. 30)). Because reducing actual

hydrocarbons to $\text{CH}_{\text{H/C}}$ is only a measure of dividing molecules into smaller ones, the difference is only important for gas properties given per mole. In a mass quantity, the amounts of carbon and hydrogen are still the same. $\text{CH}_{\text{H/C}}$ is named C_xH_y in the composition vector. For hydrogen, no H/C ratio is specified, because no carbon is assumed to be present.

To calculate the reactant enthalpy, the c_p of the fuels has to be known. The assumption is made that the slope of the c_p -polynomial (as a function of temperature) of jet fuels and diesel is the same as that of (liquid) Jet-A. The slope of the c_p -polynomial of natural gas and hydrogen is assumed to be the same as that of methane and pure hydrogen, respectively. The slopes of methane, hydrogen and liquid Jet-A are taken from the NASA CEA-program. To allow for (small) differences in the fuel, the absolute value of c_p is determined by the user: the user specifies a c_p value at a certain (user specified) temperature, and the c_p -polynomial is moved upwards or downwards. For jet fuels and diesel, this temperature must be between 220 and 550 (K) while for other fuels it must be between 200 and 6000 (K). Now that the c_p -polynomial is known, the reactant enthalpy can easily be calculated.

The determination of the heat of reaction for these fuels is less easy than for fuels with user specified composition. It is calculated using the fuel lower heating value, which is to be given by the user. The lower heating value at a certain temperature is defined as the heat released if a reactant (at the given temperature) is completely burnt and the products are cooled to the reactant temperature, where water is assumed to stay in the vapour phase. In case of complete burning, the lower heating value could be directly used as the heat of reaction. However, in gas turbine combustion chambers, combustion is seldom (locally) complete because of fuel-rich zones and dissociation. This causes the heat release to be lower than the heating value. Therefore, the lower heating value is converted into a heat of formation of the reactant ($\text{CH}_{\text{H/C}}$ or H_2) at 298.15 (K).

This is done in two steps. Because the lower heating value is specified at a reference temperature (the same temperature as where the c_p is specified, see above), that is not necessarily 298.15 (K), the first step is converting the given heating value to the heating value at 298.15 (K). This can be done using the fact that at constant pressure (or volume), the heat released by a process is independent on the path chosen (see appendix B). In figure J.3, it is shown that the change from state A to state B can be made by letting the reaction take place at T_1 and subsequently heating the products to T_2 , but also by heating the reactants to T_2 and letting the reaction take place at T_2 . The resulting heat change must be the same for both reaction paths. Therefore, the following equation relates the heat of reaction at two different temperatures:

$$H_{\text{reactants}} + \Delta H_r(T_1) = \Delta H_{\text{products}} + \Delta H_r(T_2) \quad (\text{J.1})$$

where: $\Delta H_r(T)$ = Heat of reaction at temperature T (J/kg).

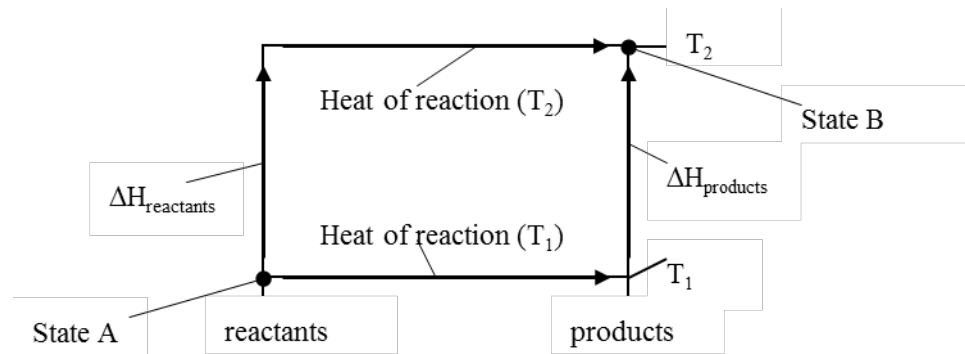


Figure J.3 Different temperature paths leading to the same state (Kuo, Ref. 26)

Because the lower heating value assumes complete combustion, the products are carbon dioxide and water in a ratio determined by the H/C-ratio. The enthalpy change involved in heating the reactants is calculated in the same way as the reactant enthalpy (see above). The lower heating value at 298.15 (K) is calculated using equation (J.1).

The second step is calculating the heat of formation of the fuel from the lower heating value at 298.15 (K). This can easily be done, because the lower heating value is equal to the heat of reaction if the combustion is complete. Because the heat of reaction is the summation of formation enthalpies, and because the formation enthalpies of water (vapour) and carbon dioxide are known, the heat of formation of the reactant (fuel) can be calculated. Now, the heat of reaction in situations of non-complete combustion can simply be calculated as the difference between summations of formation enthalpies.

For both categories of fuel, the needed enthalpy is calculated in exactly the same way as described in appendix H, by heating the products to the equilibrium temperature. Here, the C_xH_y component is assumed zero, so no unknown c_p -polynomial is needed. The assumption that no significant amounts of hydrocarbons will be present in the equilibrium products is reinforced by (Holderness, Ref. 18), stating that hydrocarbon equilibrium fractions are usually negligible for equivalence ratios lower than 3 (or even higher). The equivalence ratios in gas turbines will usually not exceed 1.8.

J.2 Calculating equilibrium at a given temperature

The method used to calculate the equilibrium composition at a given temperature in the combustion chamber is similar to the method used to calculate the equilibrium composition in case of dissociation in the gas model. However, because the temperatures in the combustion chamber are assumed to be higher than the temperatures in other components, more species will have significant fractions in the equilibrium compositions. Therefore, O, H, and OH are added to the fractions to be calculated. Because the NO_x emission model needs the NO and N₂O equilibrium fractions, N₂, NO and N₂O equilibrium concentrations are also calculated. Because of these six extra species, six extra equations are needed (to be added to equations (H.7) to (H.11)). The nitrogen atom balance forms one equation:

$$(2n_{N_2} + 2n_{N_2O} + n_{NO})_{before} = (2n_{N_2} + 2n_{N_2O} + n_{NO})_{after} \quad (J.2).$$

The five additional equations are formed by five extra equilibrium constants added:

$$K_{p,OH} = \frac{n_{OH}}{n_O n_H} \left(\frac{p}{N} \right)^{-1} \quad (J.3),$$

$$K_{p,H} = \frac{n_H}{\sqrt{n_{H_2}}} \sqrt{\frac{p}{N}} \quad (J.4),$$

$$K_{p,O} = \frac{n_O}{\sqrt{n_{O_2}}} \sqrt{\frac{p}{N}} \quad (J.5),$$

$$K_{p,NO} = \frac{n_{NO}}{\sqrt{n_{N_2} n_O}} \left(\frac{p}{N} \right)^{-\frac{1}{2}} \quad (J.6),$$

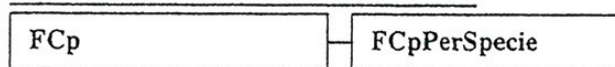
$$K_{p,N_2O} = \frac{n_{N_2O}}{n_{N_2} n_O} \left(\frac{p}{N} \right)^{-1} \quad (J.7).$$

These equilibrium constants are not based on actual (significant) reactions. Because the equilibrium state is assumed to be reached, all the reactions are in equilibrium, so it doesn't matter what reaction is used.

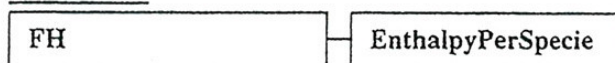
The iterative procedure used to find the equilibrium composition is similar to the one used to find the dissociation equilibrium composition. It starts with the guess of a fraction. For this guess, the other fractions are calculated and a new value for the guessed fraction results. The deviation between the guessed value for the fraction and the new value found, is used to determine the correct value for the fraction. Once this correct value is found, the equilibrium composition is also found. Because the guessed fraction must be non-zero under all circumstances to enable solving the system of equations, the O_2 -fraction is guessed. An additional advantage of choosing the O_2 -fraction is that the O-fraction can be directly calculated using equation (J.5). The O-fraction couples equation (J.3), (J.6) and (J.7). Therefore, the system is partially decoupled, and less effort is needed to solve the equations.

However, the choice of the O_2 -fraction as the guessed fraction also entails a problem. Standard Delphi numbers have a limited accuracy. This gives problems if the fractions are lower than 1.10^{-15} . Therefore, the above method can only be used as long as the O_2 -fraction is higher than 1.10^{-15} . Usually, this is no problem, but for very rich mixtures (equivalence ratios higher than approximately 1.7, depending on temperature) the O_2 -fraction can become smaller, especially if the temperature is low. To solve this problem, special numbers could be used in Delphi for the concentrations. This would result in higher accuracy (in the order of 1.10^{-30}), but also in longer computation times. Because in all practical cases, the O_2 -fraction is assumed to be higher than 1.10^{-15} , standard numbers are used.

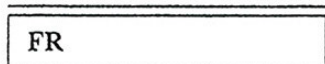
Specific heat at constant pressure:



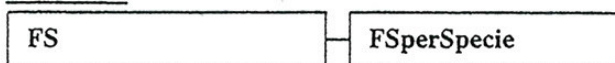
Enthalpy:



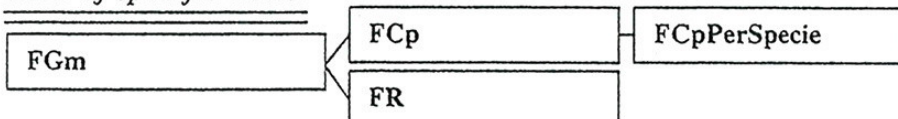
Specific Gas Constant:



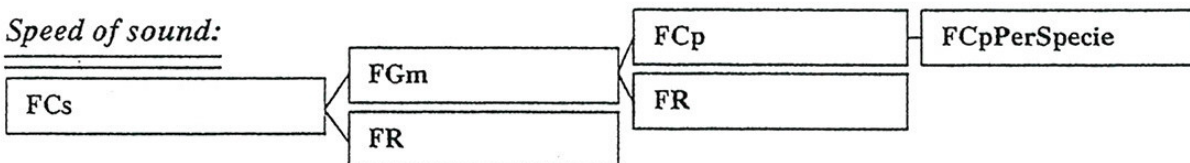
Entropy:



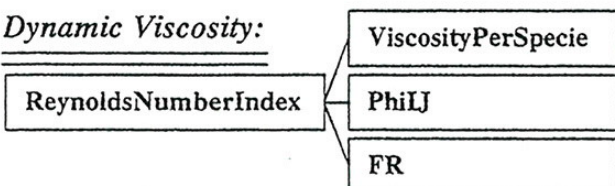
Ratio of specific heats:



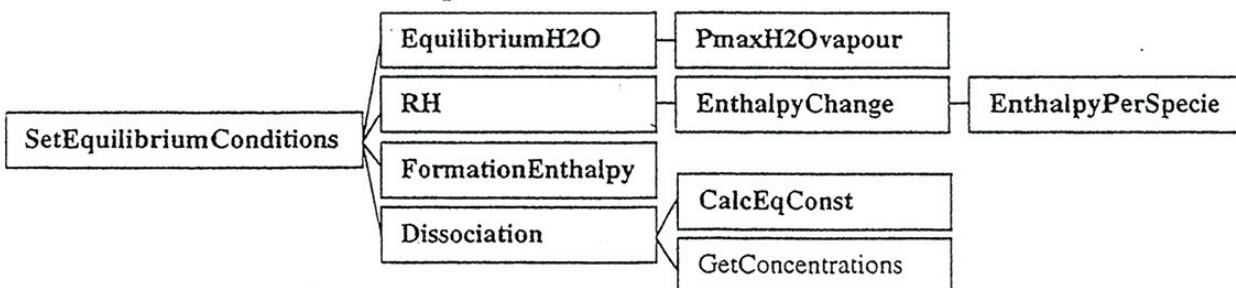
Speed of sound:



Dynamic Viscosity:



Functions used to calculate equilibrium:



Appendix K Delphi code of the gas and combustor model

The new functions and procedures of the gas and combustor model are placed in the units 'Combusn.pas' and 'GSPglobal.pas'. In Combusn.pas, procedures and functions only used in the combustion chamber are present. GSPglobal.pas is a unit that is used in all the components. Therefore, all the function and procedures to be used in the new gas model are placed in GSPglobal.pas. It is noted that the Delphi-code listed in this report is the initial code as developed by Kluitters. Current GSP code can be modified and extended.

K.1 Structure of the gas and combustor model

In this paragraph, the relations between the functions used by the gas and combustor model are shown. The structure diagrams presented in this chapter give the relations between the major part of the new functions. However, not all the functions and relations are shown in these diagrams to enable better overview. There are five important functions that are not shown because they are used very often throughout the complete gas and combustor model. They convert mass fractions into mole fractions or volume fractions and the other way around. These functions are:

- molecomposition,
- masscomposition,
- volumecomposition,
- HCratiomasscomposition,
- HCratiomolecomposition.

Gas model

In figure K.1, the structure of the gas model is shown. To start with, the functions used to calculate the thermodynamic properties are depicted.

- Calculating the specific heats for the species (using equation (C.22)) by the function *FCpPerSpecie* and subsequently taking the weighted average (using equation (C.23)) by *FCp* achieve the calculation of the specific heat at constant pressure.
- The enthalpy is calculated in the same way. *EnthalpyPerSpecie* is used to find the enthalpy for the separate species, using equation (C.25), and *FH* for the enthalpy of the complete mixture using equation (C.26).
- The specific gas constant is calculated by function *FR*, using equation (H.1).
- The entropy is calculated by the function *FS* using equations (C.32) and (C.33) and the function *FSperSpecie* calculating the steady state entropy using equation (C.34).

- The ratio of specific heats (γ) is calculated by function *FGm* applying equation (C.10) using the functions for the specific heat at constant pressure and the specific gas constant.

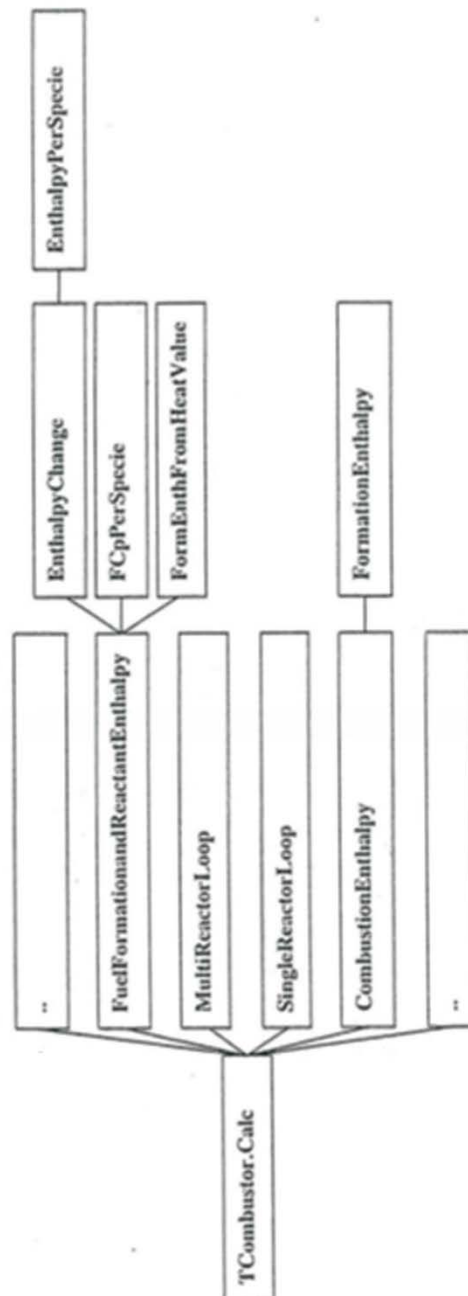


Figure K.2 Pascal code structure of the new parts of the combustor model

The speed of sound is determined using the function *FCs*, that uses the ratio of specific heats and specific gas constant according to equation (C.11).

- There is no procedure to explicitly calculate the dynamic viscosity. However, it is calculated within the procedure *ReynoldsNumberIndex*, using equations (C.41) to find the dynamic viscosity and (I.9) to calculate the Reynolds Number Index. The function *ViscosityPerspecie* applies equation (C.40) to find the viscosity per specie and *PhiIJ* uses equation (C.43) to find the viscosity interaction parameter. Because equation (I.9) contains the specific gas constant, the function *FR* is also used.

Also in figure K.1, the procedures determining the equilibrium composition and temperature are shown. First of all, the function *EquilibriumH2O* determines the equilibrium between liquid water and water vapour at a given temperature, by comparing the maximum partial water vapour pressure determined by *PmaxH2Ovapour* with the vapour pressure that would result if all the water would be vapour. To find the correct temperature together with the correct equilibrium composition, the method described in paragraph H.3 is used. To calculate the enthalpies depicted in figure H.2, the procedures *RH* and *FormationEnthalpy* are used. *RH* determines enthalpy changes involved in temperature changes from or to the chemical reference temperature of $T = 298.15$ (K) (Reactant enthalpy and Needed enthalpy), while *FormationEnthalpy* is used to find the heat of reaction (by summation of formation enthalpies). If the temperatures are higher than 1800 (K), the function *Dissociation* is used to find the equilibrium composition at a given temperature, taking dissociation into account to a limited extent. The assumption is made that it is approximately valid until $T = 2200$ (K). This function uses *CalcEqConst* to calculate the value of equilibrium constants. The procedure *GetConcentrations* is (only) locally used within *Dissociation*.

Combustor model

In figure K.2, the structure of the combustor model is shown. The central procedure coordinating the combustion calculations is *TCombustor.Calc*. This procedure is not shown here, because it was already present in GSP 7.0. *TCombustor.Calc* uses four new functions. The points in the boxes denote old functions (that are still used).

The function *FuelFormationandReactantEnthalpy* is used to determine the heat of formation and reactant enthalpy for fuels whose exact composition is not known. The function *FCpPerSpecie* is used to calculate the difference between the user specified specific heat at constant pressure and the standard value for the fuel. Using this difference, the specific heat at constant pressure is found as a function of temperature (see paragraph J.1). This is used by the function *EnthalpyChange* to calculate the reactant enthalpy and the enthalpy involved in a

temperature change of the reactants, needed to find the heating value at the chemical reference temperature ($T = 298.15$ (K)). The function *FormEnthFromHeatValue* is subsequently used to

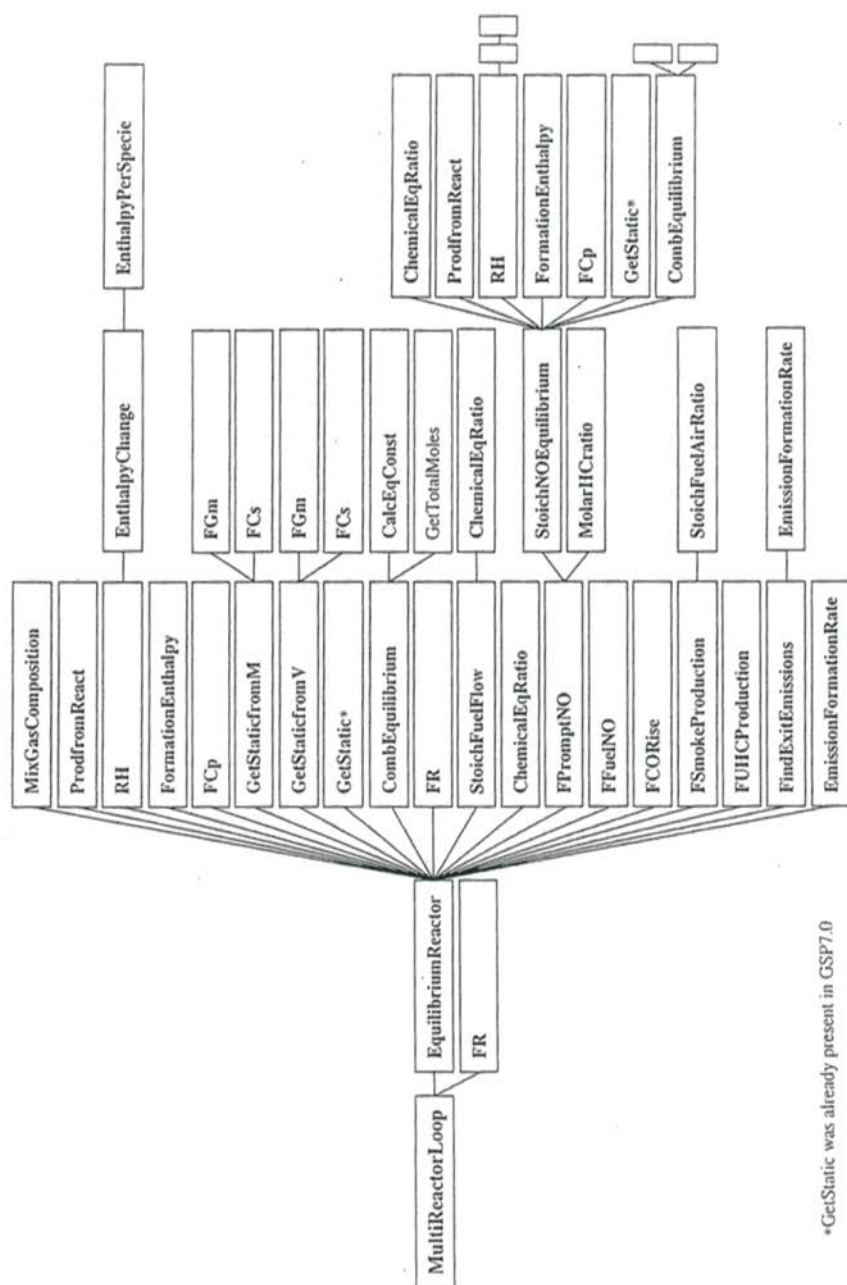


Figure K.3 Pascal code structure of the function 'MultiReactorLoop'

*GetStatic was already present in GSP7.0

calculate the heat of formation from the lower heating value at the chemical reference temperature.

For each combustion chamber in a gas turbine model, the procedure *MultiReactorLoop* is called. This procedure, depicted in figure K.3, uses the procedure *EquilibriumReactor* for each reactor in the combustor model. Within *EquilibriumReactor*, the equilibrium temperature and composition as well as the emission levels are calculated. The function *FR* is used to calculate the density, used for conversions, from pressure and temperature (using the ideal gas law). In *EquilibriumReactor*, first a guess of the equilibrium composition is made using *MixGasComposition* in case there is no fuel added and otherwise using *ProdfromReact*. *MixGasComposition* simply mixes compositions assuming no reactions to occur. *ProdfromReact* assumes complete combustion if enough oxygen is present. Otherwise the fuel reacts to CO and H₂ and eventual oxygen left is used to partly oxidise CO and H₂. No dissociation is modelled here. Like in the gas model, the combination of *RH* and *FormationEnthalpy* is used to calculate the relevant enthalpies involved in chemical reactions (see figure H.2). The specific heat at constant pressure c_p is only used in the guess for the temperature.

Once the temperature and composition guesses are found, the actual equilibrium temperature and composition are found using *RH* and *FormationEnthalpy* again for the enthalpy changes for the chemical reactions. Only now, *CombEquilibrium* is used to find the equilibrium composition at given temperature (and pressure). *CombEquilibrium* takes dissociation into account, and can be used for temperatures higher than the maximum temperature of 2200 (K) for *Dissociation*. Like *Dissociation*, *CombEquilibrium* uses *CalcEqConst* to calculate the equilibrium constants. *CombEquilibrium* comprises a function only used there: *GetTotalMoles*. Here, a difference is made between the total and static temperatures and pressures using *GetStatic*, in case the cross-sectional area is specified, *GetStaticfromM* in case of Mach-number specified and *GetStaticfromV* in case flow speed is specified. Both last functions use the ratio of specific heats for the relation between total and static properties and the speed of sound to find the relationship between the flow speed and Mach-number. The function *FR* is again only used to find the density of the flow.

Once the equilibrium composition and temperature are found, the emission levels can be found. The procedures *FPromptNO*, *FFuelNO*, *FCORise*, *FSmokeProduction* and *FUHCProduction* are used to find stepwise emission formation in the flame(s) of resp. prompt-NO_x, fuel-NO_x, CO, Soot and UHC. For calculation of prompt-NO_x formation the equivalence ratio is necessary. This is found using *StoichFuelFlow* or *ChemicalEqRatio*. Within the procedure *FPromptNO*, the mole fraction of hydrocarbon species is found using *MolarHCratio* and the stoichiometric equilibrium NO-concentration is calculated by a separate procedure *StoichNOEquilibrium*. Within this procedure, *ChemicalEqRatio* is used to find the fuel-flow that

makes the mixture stoichiometric. *RH*, *FormationEnthalpy*, *ProdfromReact*, *FCp*, *GetStatic* and *CombEquilibrium*

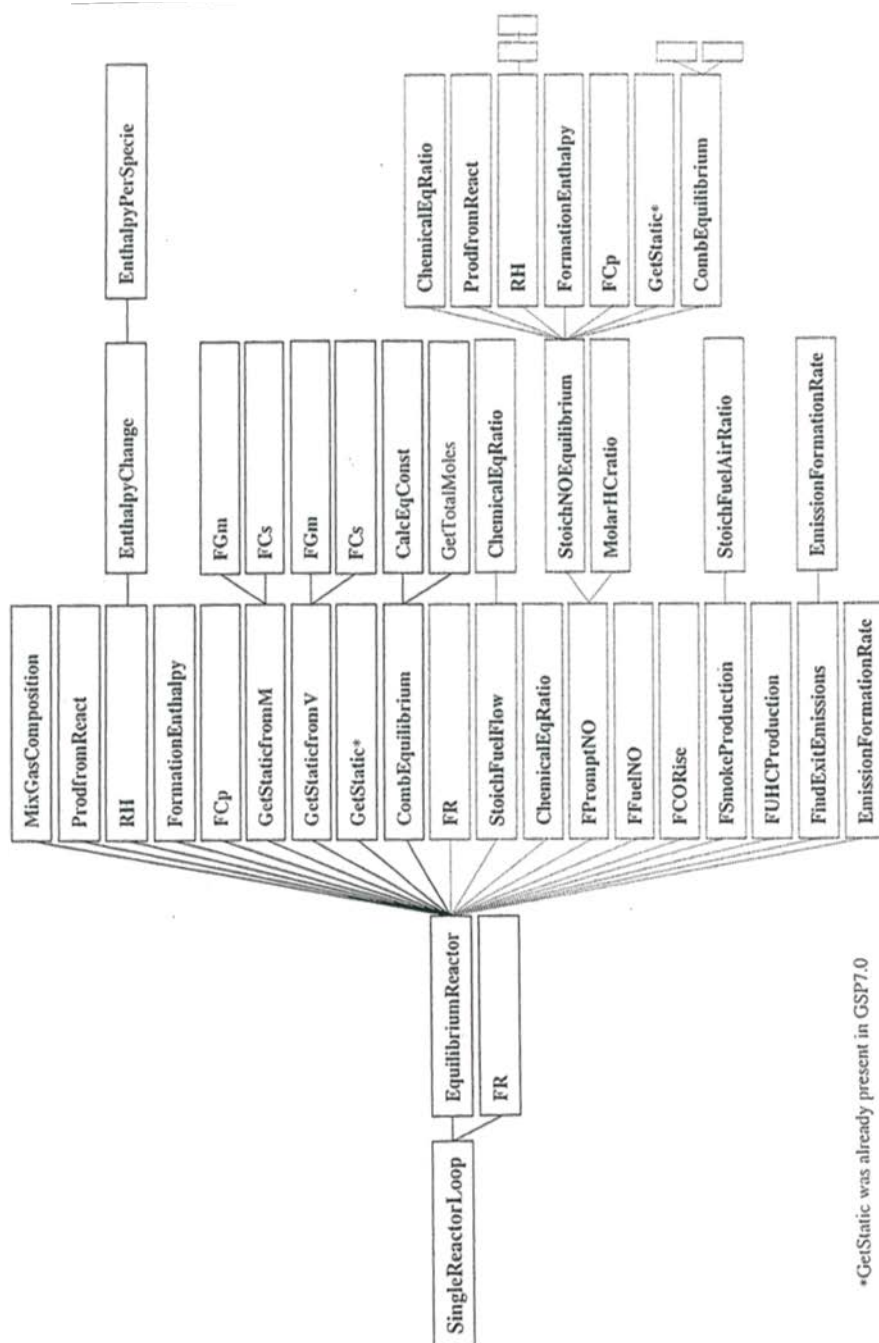


Figure K.4 Pascal code structure of the function 'SingleReactorLoop'

are, like in *EquilibriumReactor*, used to find an equilibrium guess and the actual equilibrium composition and temperature. In *FindExitEmissions*, the trapezium rule is used to find the emission levels and formation rates. To achieve this, *EmissionFormationRate* is used. In this

procedure, the emission formation rate (time-derivative) is calculated using equilibrium concentrations, temperature, pressure and the current emission specie concentration. Because all these calculations are rather time-consuming and because a lot of cycle calculations can be necessary to find the correct values for state variables (see paragraph G.2), a separate procedure *SingleReactorLoop* was designed for calculations necessary to find the state variables. Once the state variables are found, the procedure *MultiReactorLoop*, as described above will be used. In figure K.4, the structure of *SingleReactorLoop* is shown. Like *MultiReactorLoop*, *EquilibriumReactor* is used. Only now, it is only called once: the combustion chamber is modelled as one single reactor in order to calculate the equilibrium outlet conditions. Because only one reactor is used, the emissions can't be determined. Therefore, their boxes are shown with dashed lines in figure K.4.

The function *CombustionEnthalpy* (see figure K.2) is used to find the heat of reaction of a combustion reaction using the function *FormationEnthalpy*. In other functions the heat of reaction of combustion reactions is also calculated, but there it is calculated only using *FormationEnthalpy* (i.e. without explicitly calling *CombustionEnthalpy*).

K.2 Pascal code of the new procedures

In this paragraph, the Pascal code of the (new) functions and procedures mentioned in paragraph K.1 are shown. They are put in alphabetical order to enable easier finding. Behind the name of the function (/procedure), the GSP unit where the function is placed is stated.

CalcEqConst (GSPglobal.Pas)

```
//The purpose of this function is to calculate the equilibrium constant at a
//given temperature.
function CalcEqConst(T:double;Pol:Carray9):double;
begin
  Result:=Power(10,Pol[1]*IntPower(T,-2)+
    Pol[2]*IntPower(T,-1)+
    Pol[3]*LN(T)*IntPower(T,-1)+
    Pol[4]*LN(T)+
    Pol[5]+
    Pol[6]*T+
    Pol[7]*IntPower(T,2)+
    Pol[8]*IntPower(T,3)+
    Pol[9]*IntPower(T,4));
end;
```

ChemicalEqRatio (GSPglobal.Pas)

```
//This function calculates the chemical equivalence ratio. This is simply the
//quotient of the total oxygen needed for a stoichiometric mixture and the total
//oxygen that is present in the mixture. For the difference between the chemical
//equivalence ratio and the more known equivalence ratio defined by the quotient
//of actual fuel-air-ratio and stoichiometric fuel-air-ratio, readers are
//referred to NASA RP1311, Users Manual. However, a few remarks are made here:
//if all the positive valence atoms (C,H,..) are present in the fuel and all the
//negative valence atoms (O,..) in the oxidator, the two equivalence ratios are
//equal. If not, they are still equal when they are one (stoichiometric
//mixture), and they are both smaller than one for lean mixtures and higher than
```

```
//one for rich mixtures. The chemical equivalence ratio can be determined for a
//mixture, without prior knowledge of the fuelcomposition, for the other
//equivalence ratio, the fuelcomposition must be known.
function ChemicalEqRatio(const GasComposition: TGasComposition;
                        const HCRatio      : Double) : Double;
var
    O2needed, O2available, CinMixture, HinMixture, OinMixture      : Double;
    GasComp                                                    :
TGasComposition;
begin
    GasComp:=HCRatioMoleComposition(atomweightC+atomweightH*HCRatio, GasComposition);
    CinMixture:=GasComp[gtCO2]+GasComp[gtCO]+GasComp[gtCH4]+2*GasComp[gtC2H6]
                +2*GasComp[gtC2H4]+3*GasComp[gtC3H8]+4*GasComp[gtC4H10]+GasComp[gtCxHy];
    HinMixture:=2*GasComp[gtH2Og]+2*GasComp[gtH2]+GasComp[gtH]+GasComp[gtOH]
                +4*GasComp[gtCH4]+6*GasComp[gtC2H6]+4*GasComp[gtC2H4]
                +8*GasComp[gtC3H8]+10*GasComp[gtC4H10]+HCRatio*GasComp[gtCxHy];
    OinMixture:=2*GasComp[gtCO2]+GasComp[gtCO]+2*GasComp[gtO2]+GasComp[gtOH]+GasComp[gtO]+
                +GasComp[gtH2Og]+GasComp[gtNO]+GasComp[gtN2O];
    O2Needed:=CinMixture+0.25*HinMixture;
    O2Available:=0.5*OinMixture;
    Result:=O2Needed/O2Available;
end;
```

CombEquilibrium (GSPglobal.Pas)

```
//The aim of this function is to find the equilibrium composition at a given
//temperature and (static) pressure. Equilibrium is determined for (CO2, CO, O2,
//(Ar,) H2O(g), H2, O, H, OH, NO, N2O, N2). It is only valid at temperatures at
//which liquid water is negligible. In the equilibrium composition, hydrocarbons
//are assumed not to be present, which is the case until very rich
//(E.R. > 3) mixtures.
//To find the equilibrium composition, an initial guess is made of a
//concentration. Because the number of unknowns (concentrations) is equal to the
//number of equations, the other unknowns can be calculated for this guess, and
//a new value is found for the guessed concentration. The correct concentration
//is found if the new value for the concentration is equal to the guessed one.
//Therefore, the deviation of the new value from the guessed one is used to find
//the correct concentration and therefore the correct the equilibrium
//composition.
//The choice of the concentration to be guessed is determined by the fact that
//it must be a non-zero (not below 1e-15) concentration under all circumstances,
//otherwise, the equations can't be solved. Species containing C and H-atoms
//can't be chosen, because it must be possible to burn hydrogen in air (not
//containing carbondioxide) and to burn fuels only containing carbon (e.g. pure
//carbonmonoxide). In those cases, the concentration would be zero. Taking
//nitrogen or argon leads to problems if not air is used, but pure oxygen.
//Therefore, the most logical concentration is the O2-concentration, because it
//will always be present in gas turbine combustion processes. However, problems
//will be encountered when the O2 concentration becomes smaller than 1e-15. This
//problem could be solved by increasing the accuracy of the Delphi-calculations.
//However, this will take more calculation time, and O2-concentrations lower
//than 1e-15 usually occur only for (very) rich mixtures (E.R.> about 1.7) or
//low temperatures, which are not likely to occur in gas turbine combustion.
function CombEquilibrium(const T, p, HCRatio: double;
                        const StartGasComp:TGasComposition;
                        var Gascompositionv,
                            Gascompositionm:TGasComposition): boolean;

const InitStepSizeFactor = 0.1;
var
    nO2infirstOld, nO2insecond, nO2outfirst, nO2infirst,
    nO2outsecond, nO2guess,
    X, notchanged, CinMixture, HinMixture, OinMixture,
    NinMixture, derivative, pbar, nO2new,
    nO, nH, nOH,
    nNO, nN2O, nN2, nCO2, nCO, nH2Og, nH2, N1,
    N2, CO2eq, H2Oeq, OHeq, Oeq, Heq, NOeq, N2Oeq      :double;
    StepSizeFactor                                     :double;
    i,j                                                  :integer;
    GCmole                                              :TgasComposition;
```

```

//This (sub)function determines the composition and total number of moles for
//a given pressure and a guessed O2 starting number of moles (nO2in) as well
//as a guessed total number of moles.
function GetTotalMoles(const Nold, nO2in: Double; var Nresult : Double): Boolean;
var
    CO2quotient, H2Oquotient, OHquotient, Hquotient, Oquotient,
    NOquotient, N2Oquotient : Double;
begin
    Result:=false;
    if nO2in<0 then
        begin
            MessageDlg('nO2in<0 in'+#13+'in combustion model',mtError, [mbOK], 0);
            Exit;
        end;
    CO2quotient:=sqrt(nO2in*pbar/Nold)/CO2eq; //nCO2/nCO (n=number
    H2Oquotient:=sqrt(nO2in*pbar/Nold)/H2Oeq; //nH2O/nH2 of moles)
    Oquotient:=sqrt(Nold/pbar)*Oeq; //nO/sqrt(nO2)
    Hquotient:=sqrt(Nold/pbar)*Heq; //nH/sqrt(nH2)
    OHquotient:=pbar/Nold*OHeq; //nOH/(nO*nH)
    NOquotient:=sqrt(pbar/Nold)*NOeq; //nNO/(nO*sqrt(nN2))
    N2Oquotient:=pbar/Nold*N2Oeq; //nN2O/(nO*nN2)
    nCO:=CinMixture/(1+CO2quotient);
    nCO2:=CinMixture-nCO;
    nN2:=sqrt((- (sqrt(nO2in)*Oquotient*NOquotient)
        +sqrt(nO2in*sqr(Oquotient*NOquotient)
        +4*NinMixture*(2+2*sqrt(nO2in)*Oquotient*N2Oquotient)))
        /(4+4*sqrt(nO2in)*Oquotient*N2Oquotient));
    nN2O:=nN2*N2Oquotient*sqrt(nO2in)*Oquotient;
    nNO:=sqrt(nN2*nO2in)*Oquotient*NOquotient;
    nH2:=sqrt((- (Hquotient+Hquotient*sqrt(nO2in)*Oquotient*OHquotient)
        +sqrt(sqr(Hquotient+Hquotient*sqrt(nO2in)*Oquotient*OHquotient)
        +4*(2+2*H2Oquotient)*HinMixture))/(4+4*H2Oquotient));
    nH:=sqrt(nH2)*Hquotient;
    nO:=sqrt(nO2in)*Oquotient;
    nOH:=nO*nH*OHquotient;
    nH2Og:=nH2*H2Oquotient;
    Nresult:=notchanged+nCO2+nCO+nO2in+nH2Og+nH2+nO+nH+nOH+nNO+nN2O+nN2;
    Result:=true;
end;

begin
    Result:=false;
    StepSizeFactor:=InitStepSizeFactor;
    nO2guess:=1.0;
    pbar:=p/100000; //Procedure uses pressure in (bar)
    N2:=1;
    GCmole:=HCratioMoleComposition(atomweightC+atomweightH*HCratio, StartGasComp);
    if GCmole[gtH2O1]>NearlyZero then //All liquid water is evaporated.
        begin
            GCmole[gtH2Og]:=GCmole[gtH2Og]+GCmole[gtH2O1];
            GCmole[gtH2O1]:=0;
        end;

    //Determine the amount of C, H, O and N and notchanging species (only Ar).
    notchanged:=GCmole[gtAr];
    CinMixture:=GCmole[gtCO2]+GCmole[gtCO]+GCmole[gtCH4]+2*GCmole[gtC2H6]
        +2*GCmole[gtC2H4]+3*GCmole[gtC3H8]+4*GCmole[gtC4H10]+GCmole[gtCxHy];
    HinMixture:=2*GCmole[gtH2Og]+2*GCmole[gtH2]+GCmole[gtH]+GCmole[gtOH]
        +4*GCmole[gtCH4]+6*GCmole[gtC2H6]+4*GCmole[gtC2H4]
        +8*GCmole[gtC3H8]+10*GCmole[gtC4H10]+HCratio*GCmole[gtCxHy];
    OinMixture:=2*GCmole[gtCO2]+GCmole[gtCO]+2*GCmole[gtO2]+GCmole[gtOH]+GCmole[gtO]+
        +GCmole[gtH2Og]+GCmole[gtNO]+GCmole[gtN2O];
    NinMixture:=2*GCmole[gtN2]+GCmole[gtNO]+2*GCmole[gtN2O];

    // Calculation of equilibrium constants:
    if T>1000 then //and T<6000 (K)
        begin
            CO2eq:=CalcEqConst(T,Pol2CO2);
            H2Oeq:=CalcEqConst(T,Pol2H2O);
            Oeq:=CalcEqConst(T,Pol2O);

```

```

    Heq:=CalcEqConst(T,Pol2H);
    OHeq:=CalcEqConst(T,Pol2OH);
    NOeq:=CalcEqConst(T,Pol2NO);
    N2Oeq:=CalcEqConst(T,Pol2N2O);
end
else //T between 200 (K) (647.29 (K)) and 1000 (K)
begin
    CO2eq:=CalcEqConst(T,Pol1CO2);
    H2Oeq:=CalcEqConst(T,Pol1H2O);
    Oeq:=CalcEqConst(T,Pol1O);
    Heq:=CalcEqConst(T,Pol1H);
    OHeq:=CalcEqConst(T,Pol1OH);
    NOeq:=CalcEqConst(T,Pol1NO);
    N2Oeq:=CalcEqConst(T,Pol1N2O);
end;

//First a search is undertaken for a nO2infirst that gives a realistic
//nO2new-value (in other words a realistic guess for the O2 concentration).
nO2infirst:=nO2guess;
i:=0;
repeat //Iterate until the total number of moles (N) has converged.
    inc(i);
    if not GetTotalMoles(N2, nO2infirst, N1) then Exit;
    if not GetTotalMoles(N1, nO2infirst, N2) then Exit;
    nO2outfirst:=0.5*(OinMixture-2*nCO2-nCO-nH2Og-nOH-nO-nN2O-nNO);
until ((abs(N2-N1)/N1)<1e-8) or (i>30);
// until ((N2-N1)/N1)<0.01) or (i>30);
if (i>30) then
begin
    MessageDlg('TotalMoles 1st level iteration not converging',mtError,[mbOK],0);
    Exit;
end;
i:=0;
repeat //Iteration to find reasonable starting value for O2 concentration:
    inc(i);
    nO2insecond:=0.1*nO2infirst;
    j:=0;
    repeat //Iterate until the total number of moles (N) has converged.
        inc(j);
        if not GetTotalMoles(N2, nO2insecond, N1) then Exit;
        if not GetTotalMoles(N1, nO2insecond, N2) then Exit;
        nO2outsecond:=0.5*(OinMixture-2*nCO2-nCO-nH2Og-nOH-nO-nN2O-nNO);
    until ((abs(N2-N1)/N1)<1e-8) or (j>30);
// until ((N2-N1)/N1)<0.01) or (j>30);
    if (j>30) then
        begin
            MessageDlg('TotalMoles 2nd level iteration not converging',mtError,[mbOK],0);
            Exit;
        end;
    derivative:=(nO2outsecond-nO2outfirst)/(nO2insecond-nO2infirst);

    //nO2new is the guess of the correct O2 concentration that is found by
    //linearisation. As long as it is negative, the O2 guess is too big. Smaller
    //O2 guesses are tried to find a good starting value for the O2 guess.
    nO2new:=(nO2outfirst-nO2infirst*derivative)/(1-derivative);
{
    if nO2new<0 then
        begin
            nO2infirst:=nO2insecond;
            nO2outfirst:=nO2outsecond;
        end; }
    if nO2new<nO2insecond then //Decrease O2 concentration guess to find
        begin //reasonable estimate.
            nO2infirst:=nO2insecond;
            nO2outfirst:=nO2outsecond;
        end;
// until (nO2new>0) or (i>30);
until (nO2new>nO2insecond) or (i>30);
    if (nO2infirst<NearlyZero) or (i>30) then //The O2 concentration is too small
        begin //Exit procedure.
            MessageDlg('O2 concentration too small',mtError,[mbOK],0);

```

```

Exit; // exit without reporting error; leave this to calling procedure
end;
nO2infirst:=nO2new;           //End of first part of procedure CombEquilibrium.
nO2infirstOld:=nO2insecond; //First guess for O2-concentration is found.

i:=0;
repeat //Main part of the procedure determining the various concentrations.
//The same equations as in the upper part of this function are used. If
//a negative O2 concentration guess is found, a smaller O2 guess is
//applied until a positive O2 concentration guess is found.
inc(i);
j:=0;
repeat //Iterate until the total number of moles (N) has converged.
inc(j);
if not GetTotalMoles(N2, nO2infirst, N1) then Exit;
if not GetTotalMoles(N1, nO2infirst, N2) then Exit;
nO2outfirst:=0.5*(OinMixture-2*nCO2-nCO-nH2Og-nOH-nO-nN2O-nNO);
until ((abs(N2-N1)/N1)<1e-8) or (j>30);
// until (((N2-N1)/N1)<0.01) or (j>30);
if (j>30) then
begin
MessageDlg('TotalMoles 2nd level iteration not converging',mtError,[mbOK],0);
Exit;
end;

// new WV 26-8-1998 exp(... term too long )... cannot be evaluated right ???
{old if nO2infirst>nO2infirstOld then
nO2insecond:=exp(abs(StepSizeFactor*(nO2infirst-nO2infirstOld)
// (nO2infirst)))*nO2infirst
else nO2insecond:=exp(abs(StepSizeFactor*(nO2infirst-nO2infirstOld)
// (nO2infirstOld))*nO2infirst; }

//new : evaluate expression in two steps
if nO2infirst>nO2infirstOld then
X:=StepSizeFactor*(nO2infirst-nO2infirstOld)/(nO2infirst)
else X:=StepSizeFactor*(nO2infirst-nO2infirstOld)/(nO2infirstOld);
nO2insecond:=exp(abs(X))*nO2infirst;

j:=0;
repeat //Iterate until total number of moles (N) has converged.
inc(j);

//end new 31-7-1998
if not GetTotalMoles(N2, nO2insecond, N1) then Exit;
if not GetTotalMoles(N1, nO2insecond, N2) then Exit;
nO2outsecond:=0.5*(OinMixture-2*nCO2-nCO-nH2Og-nOH-nO-nN2O-nNO);
until ((abs(N2-N1)/N1)<1e-8) or (j>30);
// until (((N2-N1)/N1)<0.01) or (j>30);
if (j>30) then
begin
MessageDlg('TotalMoles 2nd level iteration not converging',mtError,[mbOK],0);
Exit;
end;

derivative:=(nO2outsecond-nO2outfirst)/(nO2insecond-nO2infirst);
nO2new:=(nO2outfirst-nO2infirst*derivative)/(1-derivative); //New start value
if nO2new<NearlyZero then

// StepSizeFactor:=StepSizeFactor*5
begin
//Decrease O2 estimate to find positive new O2 guess
nO2infirstOld:=nO2infirst; // volgens mail van Steven 2-9-1998
nO2infirst:=0.85*nO2infirst; // dit werkt dus met LM2500f !! (2-9-1998)
// MessageDlg('Negative O2 concentration in CombEquilibrium',mtError,[mbOK],0);
// Exit;
// end
else
begin
// new WV 27-8-1998
// StepSizeFactor:=InitStepSizeFactor; // reset
nO2infirstOld:=nO2infirst;
nO2infirst:=nO2new;

```

```

end;

until (abs((nO2infirstOld-nO2infirst)/nO2infirst)<0.00001) or (i>40);
if (i>40) then //To save time, the above mentioned accuracy can be lowered.
begin
  MessageDlg('No convergence in composition-iteration',mtError,[mbOK],0);
  Exit;
end;

//Calculation of new fractions:
Gascompositionv[gtCO2]:=nCO2/N2;
Gascompositionv[gtCO]:=nCO/N2;
Gascompositionv[gtO2]:=nO2new/N2;
Gascompositionv[gtAr]:=GCmole[gtAr]/N2;
Gascompositionv[gtH2Og]:=nH2Og/N2;
Gascompositionv[gtH2Ol]:=0;
Gascompositionv[gtH2]:=nH2/N2;
Gascompositionv[gtCH4]:=0;
Gascompositionv[gtC2H6]:=0;
Gascompositionv[gtC2H4]:=0;
Gascompositionv[gtC3H8]:=0;
Gascompositionv[gtC4H10]:=0;
Gascompositionv[gtO]:=nO/N2;
Gascompositionv[gtH]:=nH/N2;
Gascompositionv[gtOH]:=nOH/N2;
if NinMixture=0 then //Because of very small numerical mistakes, the concen-
begin //trations of N-containing species could be non-zero.
  Gascompositionv[gtNO]:=0;
  Gascompositionv[gtN2O]:=0;
  Gascompositionv[gtN2]:=0;
end
else
begin
  Gascompositionv[gtNO]:=nNO/N2;
  Gascompositionv[gtN2O]:=nN2O/N2;
  Gascompositionv[gtN2]:=nN2/N2;
end;
Gascompositionv[gtCxHy]:=0;
Gascompositionm:=MassComposition(Gascompositionv);
Result:=true;
end; {end CombEquilibrium}

```

CombustionEnthalpy (GSPglobal.Pas)

```

//This function calculates the heat of reaction of a combustion reaction (change
//in formation enthalpies due to the reaction).
function CombustionEnthalpy(const Wfuel, Woxid : Double;
                             const FuelComp, OxidComp, ProdComp: TGasComposition
                             ): Double;

begin

Result:=      Wfuel*FormationEnthalpy(FuelComp)
              +Woxid*FormationEnthalpy(OxidComp)
              -(Wfuel+Woxid)*FormationEnthalpy(ProdComp);

end;

```

Dissociation (GSPglobal.Pas)

```

//This function determines the equilibrium composition at a given temperature,
//just like the above function CombEquilibrium. Only here, the concentrations of
//O, H and OH are assumed negligible, which is a good approximation for
//temperatures lower than 2200 (K). Further, all Nitrogen containing species
//remain frozen. In other words, only dissociation of CO2 and H2O to CO, H2 and
//O2 is calculated. All water present is assumed to evaporate: if dissociation
//becomes important (T>1800 (K)), the temperature is higher than the critical
//temperature of H2O. Here, just like in CombEquilibrium the O2 concentration
//is guessed, a new value for the O2 concentration is found and a new guess is
//made until the correct O2 concentration (and thus the complete composition) is
//found.

```



```

function Dissociation(var Gascond : TGasConditions) : Boolean;
var
    notchanged, CinMixture, HinMixture, OinMixture, derivative,
    CO2quotient, H2Oquotient, nO2new, nO2new1, nO2infirst,
    nO2insecond, nO2outfirst, nO2outsecond, nO2guess,
    nCO2, nCO, nH2Og, nH2, N, CO2eq, H2Oeq, pbar      :double;
    i,j                                                  :integer;
    GCmole                                              :TgasComposition;

//This (sub)procedure calculates the O2 concentration from the conservation
//of O-atoms using a guess for O2. Unlike in CombEquilibrium subfunction
//GetTotalMoles the total number of moles is directly known if the O2 guess is
//known.
procedure GetConcentration(const nO2in : Double; var nO2out : Double);
begin
    N:=notchanged+CinMixture+0.5*HinMixture+nO2in;
    CO2quotient:=sqrt(nO2in*pbar/N)/CO2eq; //nCO2/nCO          (n = number
    H2Oquotient:=sqrt(nO2in*pbar/N)/H2Oeq; //nH2O/nH2          of moles)
    nCO:=CinMixture/(1+CO2quotient);
    nCO2:=CinMixture-nCO;
    nH2:=0.5*HinMixture/(1+H2Oquotient);
    nH2Og:=0.5*HinMixture-nH2;
    nO2out:=0.5*OinMixture-nCO2-0.5*nCO-0.5*nH2Og;
// new v.8.0.0.9 WV 20-10-1998 :
// if nO2out<0 then nO2out:=0;
    end;
begin
    Result:=false;
    nO2guess:=1.0;
    pbar:=Gascond.Pt/100000; //The procedure works with pressures in (bar).
    i:=0;
    j:=0;
    GCmole:=MoleComposition(Gascond.Composition);
    if GCmole[gtH2O1]>NearlyZero then //Evaporate present water.
    begin
        GCmole[gtH2Og]:=GCmole[gtH2Og]+GCmole[gtH2O1];
        GCmole[gtH2O1]:=0;
    end;

    //Calculate amounts of C, H, O and notchanging species present.
    //Allow frozen fractions of NO and N2O !
    notchanged:=GCmole[gtAr]+GCmole[gtNO]+GCmole[gtN2]+GCmole[gtN2O];

    CinMixture:=GCmole[gtCO2]+GCmole[gtCO];
    HinMixture:=2*GCmole[gtH2Og]+2*GCmole[gtH2]+GCmole[gtOH]+GCmole[gtH];
    OinMixture:=2*GCmole[gtCO2]+GCmole[gtCO]+2*GCmole[gtO2]
        +GCmole[gtH2Og]+GCmole[gtOH]+GCmole[gtO];

    //Calculate the equilibrium constants.
    CO2eq:=CalcEqConst(Gascond.Tt,Pol2CO2); //Only temperatures above 1000 (K)!
    H2Oeq:=CalcEqConst(Gascond.Tt,Pol2H2O);

    //Iteration to find reasonable starting value for nO2:
    nO2infirst:=nO2guess;
    GetConcentration(nO2infirst, nO2outfirst);
    repeat //Find for given (first estimate of) nO2: N, nCO2, nCO, nH2O and nH2;
        inc(j);
        nO2insecond:=0.1*nO2infirst;

        GetConcentration(nO2insecond, nO2outsecond);

        derivative:=(nO2outsecond-nO2outfirst)/(nO2insecond-nO2infirst);

        //nO2new is the guess of the correct O2 concentration that is found by
        //linearisation. As long as it is negative, the O2 guess is too big. Smaller
        //O2 guesses are tried to find a good starting value for the O2 guess.
        nO2new:=(nO2outfirst-nO2infirst*derivative)/(1-derivative);
        if nO2new<0 then //Decrease O2 concentration guess to find
            begin //reasonable estimate.
                nO2infirst:=nO2insecond;
                nO2outfirst:=nO2outsecond;
            end;
        end;
    until nO2new>0;
    nO2out:=nO2outfirst;
    Result:=true;
end;

```

```

        end;
until (nO2new>0) or (j>15);
if (j>15) then      //The O2 concentration is too small because of limited
begin              //accuracy. Exit function.
    MessageDlg('O2 concentration too small',mtError,[mbOK],0);
    Exit;
end;
nO2infirst:=nO2new;
nO2new1:=nO2insecond;

//Main part of the procedure determining the various concentrations. The same
//formulae as in the above part are used, only now, a negative O2
//concentration guess will cause failure (see CombEquilibrium for a solution
//to this problem if thought necessary).
repeat
    inc(i);
    GetConcentration(nO2infirst, nO2outfirst);

    if nO2new>nO2new1 then
        nO2insecond:=exp(abs((nO2new-nO2new1)/(10*nO2new)))*nO2infirst
    else nO2insecond:=exp(abs((nO2new-nO2new1)/(10*nO2new1)))*nO2infirst;

    GetConcentration(nO2insecond, nO2outsecond);

    derivative:=(nO2outsecond-nO2outfirst)/(nO2insecond-nO2infirst);
    nO2new1:=nO2infirst;
    nO2new:=(nO2outfirst-nO2infirst*derivative)/(1-derivative);
    nO2infirst:=nO2new;
until (abs((nO2new1-nO2new)/nO2new)<0.00001) or (i>40);
if (i>40) then
begin
    MessageDlg('No convergence in composition-iteration',mtError,[mbOK],0);
    Exit;
end;
with Gascond do
begin      //Determine equilibrium mole fractions.
    Composition[gtCO2]:=nCO2/N;
    Composition[gtCO]:=nCO/N;
    Composition[gtO2]:=nO2new/N;
    Composition[gtAr]:=GCmole[gtAr]/N;
    Composition[gtH2Og]:=nH2Og/N;
    // Composition[gtH2O1]:=GCmole[gtH2O1]/N;    assumed 0
    Composition[gtH2]:=nH2/N;
    // Composition[gtCH4]:=GCmole[gtCH4]/N;    assumed 0
    // Composition[gtC2H6]:=GCmole[gtC2H6]/N;    assumed 0
    // Composition[gtC2H4]:=GCmole[gtC2H4]/N;    assumed 0
    Composition[gtN2]:=GCmole[gtN2]/N;
    Composition[gtN2O]:=GCmole[gtN2O]/N;
    // number of NO unchanged but concentration changed due to change in N = total moles
    Composition[gtNO]:=GCmole[gtNO]/N;
    Composition:=MassComposition(Composition);
end;
Result:=true;
end;

```

EmissionFormationRate (Combustn.Pas)

```

//This function is used to find the formation rate (=time derivative) of the
//emissions.
function TCombustor.EmissionFormationRate(const Emission : TEmission;
                                           const Gasconc: TGasComposition;
                                           const Ps, Ts, Estimate: double): double;

const SootDensity = 1800; // (kg/m3)
var    R                                           : Carray9;
      NOEquilibFraction, PartialO2Pressure, x : double;
begin
case Emission of
    etNOx : begin
        R[1] := 1.8e11*exp(-38367/Ts)*Gasconc[gtO]*Gasconc[gtN2]; //Equation: //6.3

```

```

R[2] := 1.37e6*Ts*exp(-19239/Ts)*Gasconc[gtNO]*Gasconc[gtO]; //6.4
R[3] := 8.12e10*exp(-24125/Ts)*Gasconc[gtNO]*Gasconc[gtH]; //6.5
R[4] := 7.6e10*exp(-7648.9/Ts)*Gasconc[gtN2O]*Gasconc[gtH]; //6.9
R[5] := 1.0e11*exp(-14190.7/Ts)*Gasconc[gtO]*Gasconc[gtN2O]; //6.7
R[6] := 1.0e11*exp(-14190.7/Ts)*Gasconc[gtO]*Gasconc[gtN2O]; //6.8

//Because of the third body in this reaction ('M') (the order of)
//this reaction is dependent on pressure in a way described by a
//Lindemann fall-off. Here, Lindemann fall-off is discarded.
//For pressures lower than 10 (bar), the reaction is assumed to be
//pressure dependent, for pressures higher than 10 (bar) not.
if Ps<1000000 then //arbitrary pressure: could be changed
    R[7] := 6.9e20*Power(Ts,-2.5)*exp(-32709.1/Ts)*Gasconc[gtN2O]
        *Gasconc[gtN2]
else //High pressure assumption
    R[7] := 1.3e8*exp(-29991.7/Ts)*Gasconc[gtN2O]; //Equation 6.6
        //Equation:
R[8] := 2.78648e11*exp(-30264.6/Ts)*Gasconc[gtN2O]*Gasconc[gtCO]; //6.11
R[9] := 8.439e11*exp(-16118.4/Ts)*Gasconc[gtN2O]*Gasconc[gtH]; //6.10
NOEquilibFraction:=Estimate/Gasconc[gtNO]; //=[NO]/[NO]eq
Result:=2*(1-sqr(NOEquilibFraction)) //Equation 6.20.
        *(R[1]/(1+NOEquilibFraction*R[1]/(R[2]+R[3]))
        +(R[6]+ (R[8]+R[9])/2/(1+NOEquilibFraction)
        *(1+NOEquilibFraction*R[6]/(R[4]+R[5]+R[7])))
        /(1+(R[6]+R[8]+R[9])/(R[4]+R[5]+R[7])));
end;
etCO : begin
    Result:=1.5e4*Power(Ts,1.30)*exp(3200/8.31451/Ts) //Equation 6.24.
        *-Gasconc[gtOH]*(1+Gasconc[gtCO]/Gasconc[gtCO2])
        *(Estimate-Gasconc[gtCO]);
end;
etUHC : begin
    case FuelTypeIndex of
        0, 1, 2, 3 : if Ts>555.5556 then Result:= -3.16e11
            *Power(Ps/100000,-0.815)*exp(-12200/Ts)
            *(9e-4*Ts-0.5)*sqrt(Estimate)*Gasconc[gtO2]
        else Result:=0; //Equation 6.26
        4, 6: Result:= -1.585e10*exp(-24355.69/Ts)*Power(Estimate, 0.7)
            *Power(Gasconc[gtO2], 0.8); //Equation 6.27
        5: Result := 0;
    end;
end;
etSmoke : begin //The dr/dt (r=radius) of the soot spheres is calculated;
    //the incoming emissionout is the radius at the reactor
    //entrance; gasconc consists of molefractions here.
    PartialO2Pressure:=gasconc[gtO2]*Ps/101325; //Equation:
    R[1] := 20*exp(-15096.5/Ts); //kA, p20-5 AGARD CP-125 6.32
    R[2] := 4.46e-3*exp(-7648.89/Ts); //kB, p20-5 AGARD CP-125 6.33
    R[3] := 1.51e5*exp(-48812/Ts); //kT, p20-6 AGARD CP-125 6.34
    R[4] := 21.3*exp(2063.19/Ts); //kZ, p20-6 AGARD CP-125 6.35
    x:=1/(1+R[3]/PartialO2Pressure/R[2]); //6.31
    Result:=-12/1800*10*(x*(R[1]*PartialO2Pressure
        /(1+R[4]*PartialO2Pressure))
        +R[2]*PartialO2Pressure*(1-x)); //Equation 6.30
end;
end; //end case statement
end;

```

EnthalpyChange (GSPglobal.Pas)

//The result of this function is the enthalpy change involved in a temperature
//change of a given specie.
function EnthalpyChange(const T2,T1 :Double; const GP: TGasProperties):Double;
begin
Result:=EnthalpyPerSpecie(T2, GP) - EnthalpyPerSpecie(T1, GP); //Result in [J]!
end;

EnthalpyPerSpecie (GSPglobal.Pas)

//The result of this function is the enthalpy of a specie at a given temperature
//(equation C.25). The absolute value is determined by the fact that at

```

//T = 298.15 (K) the enthalpy is equal to the formation enthalpy at
//T = 298.15 (K) of the specie.
function EnthalpyPerSpecie(const T :double; const GP: TGasProperties):double;
begin
    //CoefsChangeTemp is the bordering temperature between the
    //two sets of (NASA) coefficients (here 1000 (K)).
with GP do if T>CoefsChangeTemp then
    Result:=Gasconst/MoleMass*(-Coefs_above1000[1]*IntPower(T,-1)
        +Coefs_above1000[2]*ln(T)
        +Coefs_above1000[3]*T
        +Coefs_above1000[4]/2*IntPower(T,2)
        +Coefs_above1000[5]/3*IntPower(T,3)
        +Coefs_above1000[6]/4*IntPower(T,4)
        +Coefs_above1000[7]/5*IntPower(T,5)
        +Coefs_above1000[8]/6*IntPower(T,6)
        +Coefs_above1000[9])
else
    Result:=Gasconst/MoleMass*(-Coefs_below1000[1]*IntPower(T,-1)
        +Coefs_below1000[2]*ln(T)
        +Coefs_below1000[3]*T
        +Coefs_below1000[4]/2*IntPower(T,2)
        +Coefs_below1000[5]/3*IntPower(T,3)
        +Coefs_below1000[6]/4*IntPower(T,4)
        +Coefs_below1000[7]/5*IntPower(T,5)
        +Coefs_below1000[8]/6*IntPower(T,6)
        +Coefs_below1000[9]); // Result in [ J ]!
end;

```

EquilibriumH2O (GSPglobal.Pas)

```

//The purpose of this function is to calculate the equilibrium between liquid
//and gaseous water. If the temperature is higher than the critical temperature
//of water (647.29 (K)) all the water will be vapour. If the temperature is
//lower, the maximum partial vapour pressure of water is calculated by
//'PmaxH2Ovapour' and compared to the vapour pressure that would exist if all
//the water would be vapour to see if all the water can be vapour. If not, a
//part will be liquid.
function EquilibriumH2O(const T, p: double;
    const GasComp: TGasComposition): TGasComposition;
var
    H2Ototal, MaxVapourWater      : double;
    GCeq                          : TGasComposition;
begin
    // initial value
    GCeq:=GasComp;
    if T>TcritH2O then //Above the critical temperature no liquid water.
    begin
        GCeq[gtH2Og]:=GasComp[gtH2Og]+GasComp[gtH2Ol];
        GCeq[gtH2Ol]:=0;
    end
    else
    begin
        //als P in [N/m2] is!!! MaxVap... is ook de maximale molfractie van H2Og
        MaxVapourWater:=PmaxH2Ovapour(T)/p;
        GCeq:=MoleComposition(GasComp);
        H2Ototal:=GCeq[gtH2Og]+GCeq[gtH2Ol];
        if H2Ototal<=MaxVapourWater then //All the water can be gaseous.
        begin
            GCeq[gtH2Og]:=H2Ototal;
            GCeq[gtH2Ol]:=0;
        end
        else //Not all the water can be gaseous;
        begin //Some liquid water remains present.
            GCeq[gtH2Ol]:=(H2Ototal-MaxVapourWater)/(1-MaxVapourWater);
            GCeq[gtH2Og]:=H2Ototal-GCeq[gtH2Ol];
        end;
        GCeq:=MassComposition(GCeq);
    end;
    Result:=GCeq;
end;

```

EquilibriumReactor (Combusn.Pas)

```

//The purpose of this function is to calculate the equilibrium temperature and
//emission levels for a reactor. The assumption made is that only one fuel
//composition is used throughout the combustion chamber. If this is not the
//case, some lines have to be changed.
function TCombustor.EquilibriumReactor(const Mode : TCalcMode;
                                     //Only used for calculation of eta.
                                     const Detail : TDetailLevel;
                                     const TheVspecifier, IntersectionNr: integer;
    var Flowin, Fuelin, Oxidin, Waterin, Flowout: TGasConditions;
    const L, dNODtin, NOin, dCODtin, COin, dUHCdtin, UHCin,
        NumberOfSmokeSpheresin, SootSphereRadiusin, dSootSphereRadiusdtin,
        WTotalFuel, WTotalOxid, WTotalWater: double;
    var dNODtout, NOout, NOeqppm, dCODtout, COout, COeqppm, O2eqppm,
        dUHCdtout, UHCout,
        NumberOfSmokeSpheresout, SootSphereRadiusout, dSootSphereRadiusdtout,
        molew_div_rho, ResTime: Double): Boolean;

var
    DeltaNO, molweight, Rho, dNODtiter, NOPandf, FuelFormationEnthalpy,
    NOxTfactor, COTfactor, UHCTfactor, SmokeTfactor, Dummy,
    NOxTemperature, COTemperature, UHCTemperature, SmokeTemperature,
    FuelReactantEnthalpy, FormEnthalpyWithoutFlowout,
    StoichFuelFlowPerKgOxid, SmokeProd, Rsootsphereinit, UHCPProduction,
    Hreactant, Hformation, Hneeded, Havail, Herror, Tcorrection, err, dir, Titer,
    NOouttiter, COouttiter, EqRatio, CxHyMoleMass,
    COproduction, PromptNO, FuelNO, dNODtout1, dNODtout2,
    dCODtout1, dCODtout2                                : Double;
    ig                                                    : TGas;
    i                                                    : integer;
    CompositionWithoutFuel, LocalComposition,
    Gasconc, Gascompnewv, StartGasComp                  : TGasComposition;
    Inmix                                                : TGasConditions;
    ConvergedAllowed, CompositionFound                    : Boolean;

begin
    Result:=false;

    //First a guess of the composition is made by mixing (no fuel added) or by
    //application of ProdFromReact. Using this new composition, a (total)
    //equilibrium temperature guess is produced.
    for ig:=Low(ig) to High(ig) do
        Inmix.Composition[ig]:= ( Flowin.W*Flowin.Composition[ig]
                                +Oxidin.W*Oxidin.Composition[ig]
                                +Waterin.W*Waterin.Composition[ig] )
                                / (Flowin.W+Oxidin.W+Waterin.W);
        Inmix.W:=Flowin.W+Oxidin.W+Waterin.W;
        StartGasComp:=MixGascomposition(Inmix.Composition, Fuelin.Composition,
                                         Inmix.W, Fuelin.W);

        // Estimate Flowout.Tt based on combustion without dissociation-----
        // and try to find estimate of combustor products composition.
        if Fuelin.W<NearlyZero then
            begin
                Flowout.Composition:=Inmix.Composition;
                Compositionfound:=true;
            end
        else
            CompositionFound:=ProdFromReact(Fuelin.W, Inmix.W, HCRatio,
                                             Fuelin.Composition, Inmix.Composition, Flowout.Composition);
        if Compositionfound then
            begin
                Flowout.W:=Inmix.W+Fuelin.W;
                Hreactant:= Flowin.W*RH(Flowin.Composition, Flowin.Tt)
                           +Oxidin.W*RH(Oxidin.Composition, Oxidin.Tt)
                           +Waterin.W*RH(Waterin.Composition, Waterin.Tt)
                           +FuelReactantEnthalpyPerKg*Fuelin.W;
                FormEnthalpyWithoutFlowout:=Flowin.W*FormationEnthalpy(Flowin.Composition)
                                           +FuelFormationEnthalpyPerKg*Fuelin.W
                                           +Oxidin.W*FormationEnthalpy(Oxidin.Composition)

```



```

+Waterin.W*FormationEnthalpy(Waterin.Composition);
Hformation:=FormEnthalpyWithoutFlowout
-Flowout.W*FormationEnthalpy(Flowout.Composition);
Havail:=Hreactant+Hformation;

i:=0;
Titer:=2000;
repeat //Solve for correct temperature.
  Hneeded:=(Flowout.W)*RH(Flowout.Composition,Titer);
  Herror:=Havail-Hneeded;
  Tcorrection:=Herror/FCp(Titer,Flowout.Composition)/(Flowout.W); //unit [J]
  Titer:=Titer+Tcorrection;
  inc(i);
until (abs(Tcorrection)< 10) or (i>16);
if (i>16) then Flowout.Tt:=2000 else Flowout.Tt:=Titer;
end
else//Compostion not found in ProdFromReact (probably due to overrich mixture)
  Flowout.Tt:=2000; //Try with rough estimate
// end estimate Flowout.Tt-----

//Second the equilibrium temperature and composition are calculated using
//CombEquilibrium and one of the functions to calculate static temperature and
//pressure.

ResetAFQmem;
i:=0;
repeat
  with Flowout do if (i>0) then //In the first loop the composition is calcu-
    //lated using total temperature and pressure.
    begin
      ConvergedAllowed:=true;

      //The old composition is used, so there should be an extra
      //'repeat-until'-loop here, but it is assumed that in the last loops used
      //to find the temperature, the composition is almost constant.
      if Mode=cmDesign then //Calculate constant station cross flow area:
        begin
          case TheVspecifier of
            //0: Flow area Specified, A = column 1
            0 : ; // A already assigned A:=ReactorGrid.DValues[1,InterSectionNr+1];
            //1: Mach number Specified , calculate A (=Reactorgrid column 1)
            1 : if not GetStaticFromM(Mach, Pt, Tt, W, Composition, Ps, Ts, V,
              A) then Exit;
            //2: Flow speed (V) Specified, calculate A (=Reactorgrid column 1)
            2 : if not GetStaticFromV(V, Pt, Tt, W, Composition, Ps, Ts, Mach,
              A) then Exit;
          end;
          ReactorGrid.DValues[1,InterSectionNr+1]:=A;
        end
      else //Set A to Reactorgrid column 1 as calculated in Design point
        A:=ReactorGrid.DValues[1,InterSectionNr+1];
        //Now get static conditions from A and W, Pt, Tt:
        if not GetStatic(W, Pt, Tt, A, Composition, Ps, Ts, Mach, V,
          'Combustion reactor') then Exit;
        // new v8.1 1-9-1998 Combequilibrium only if Wf>0 !,
        //otherwise freeze composition
        if Wf>NearlyZero then
          begin
            if not CombEquilibrium(Ts, Ps, HCRatio, StartGasComp, Gascompnewv,
              Composition) then
              begin //Here, an attempt is made to remedy the bad convergence of
                //CombEquilibrium at low temperatures and rich mixtures
                //old if not ProdFromReact(1, 0, CxHyMoleMass, StartGasComp, Air,
                //Composition) then
                //new 20-10-1998
                if not ProdFromReact(1, 0, HCRatio, StartGasComp, Air, Composition) then
                  begin
                    MessageDlg('No convergence in Equilibrium Reactor', mtError, [mbOK], 0);
                    Exit;
                  end;
                  Gascompnewv:=MoleComposition(Composition);

```



```

        ConvergedAllowed:=false; //Because the composition found with
        end; //ProdfromReact is just an estimate for the
        //composition, the loop is not allowed to be converged.
    end; // new v8.1 1-9-1998 leave composition at StartComposition set at i=0
end
else //i=0: first iteration loop
begin
    ConvergedAllowed:=false;
// new v8.1 1-9-1998 Combequilibrium only if Wf>0 !, otherwise freeze composition
    if Wf>NearlyZero then
        begin
            if not CombEquilibrium(Tt,Pt,HCRatio,StartGasComp,Gascompnewv,
                                   Composition) then
                begin
                    //old ProdFromReact(1, 0, CxHyMoleMass,StartGasComp, Air, Composition);
                    //new 20-10-1998
                    if not ProdFromReact(1, 0, HCRatio,StartGasComp, Air, Composition) then
                        Gascompnewv:=MoleComposition(Composition);
                    end;
                end
            else
                begin
                    // new v8.1 1-9-1998 Combequilibrium only if Wf>0 !, otherwise freeze composition
                    Composition:=StartGasComp;
                    Gascompnewv:=MoleComposition(Composition);
                end
            end;
        end;

// new v8.1 1-9-1998 Combequilibrium only if Wf>0 !, otherwise freeze composition
    if Wf>NearlyZero then
        begin
            //Combustion efficiency only affects heat release (not gas compositions
            //yet) at this stage.
            if ModelOpt.ItemIndex=1 then //Recalculate ETA based on Out1.Tt
                begin //Read combustion efficiency map:
                    //new 8, old if not (Map as Map2in_lout).Out1(Titer-In1.Tt,In1.Delta,
                    if not (Map as Map2in_lout).Out1(FlowOut.Tt-FlowIn.Tt,FlowIn.Delta,
                    (Self.Owner as TGSPcomp).CompIDstr+' ETA comb. map', ETAmap) then Exit;
                    if Mode=cmDesign then SFeta:=ETAdes.Value/ETAmap;
                    ETA:=SFeta*ETAmap;
                end;
                Hformation:=ETA*(FormEnthalpyWithoutFlowout
                                -Flowout.W*FormationEnthalpy(Flowout.Composition));
                Havail:=Hreactant+Hformation; //Eventually apply heat loss here.
                with Flowout do Hneeded:=W*RH(Composition,Tt);
                err:=(Havail-Hneeded)/abs(Havail);

                //In case the error is small in the first step (i=0), the new temperature
                //Titer, given by AFQuir will be far to small if 'dir:=err' is used.
                if i>0 then dir:=err
                    else dir:=0.95;
                AFQcode:=AFQUIR(Flowout.Tt,err,0,0.001,dir,40,
                                'Reactor equilibrium iteration in'+#13+
                                (Owner as TGSPcomp).CompIDstr,Titer);
                //old if Titer<200 then Titer:=220; //If T<647.29, liquid water can exist,
                if Titer<In1.Tt then Titer:=In1.Tt; //which is not accounted for in the
                if Titer>6000 then Titer:=5000; //CombEquilibrium procedure.
                Flowout.Tt:=Titer;
            end
        end
    else // new v8.1 1-9-1998 Combequilibrium only if Wf>0 !, otherwise freeze
        //composition
        begin // leave Flowout.Tt at value determined in Estimate Flowout section
            AFQcode:=2; // quit iteration when Wf=0 above
        end;
    inc(i);
until ((AFQcode>1) and (ConvergedAllowed))
    or (i>100);
if (AFQcode=3) then Exit; //Error in AFQUIR iteration
if i>100 then //CombEquilibrium not converging
begin
    MessageDlg('Comb Equilibrium procedure not converging'+#13+

```

```

        (Owner as TGSPcomp).CompIDstr,mtError, [mbOK], 0);
    Exit;
end;
NOeqppm:=GasCompnewv[gtNO]*1e+6;
O2eqppm:=GasCompnewv[gtO2]*1e+6;
COeqppm:=GasCompnewv[gtCO]*1e+6;

if Detail=dlHigh then      //Continue to calculate emissions
begin                      //(only for MultiReactorLoop).
    molweight:=0;  //Conversion of molefractions to kmole/m3.
    for ig:=Low(ig) to High(ig) do
        molweight:=molweight+Gascompnewv[ig]*GasProperties[ig].MoleMass;
    with Flowout do Rho:=Ps/FR(Composition)/Ts;

    //For conversion from kMole/m3 to ppm (volume):
    molew_div_rho:=molweight/Rho;

    for ig:=Low(ig) to High(ig) do Gasconc[ig]:=Gascompnewv[ig]*Rho/molweight;

    //Calculate temperatures using tuning factors.
    NOxTfactor:=ReactorGrid.Dvalues[8,InterSectionNr+1];
    COTfactor:=ReactorGrid.Dvalues[9,InterSectionNr+1];
    UHCTfactor:=ReactorGrid.Dvalues[10,InterSectionNr+1];
    SmokeTfactor:=ReactorGrid.Dvalues[11,InterSectionNr+1];

    NOxTemperature :=Oxidn.Tt+NOxTfactor*(Flowout.Ts-Oxidn.Tt);
    COTemperature :=Oxidn.Tt+COTfactor*(Flowout.Ts-Oxidn.Tt);
    UHCTemperature:=Oxidn.Tt+UHCTfactor*(Flowout.Ts-Oxidn.Tt);
    SmokeTemperature:=Oxidn.Tt+SmokeTfactor*(Flowout.Ts-Oxidn.Tt);

    //Convert user interface nanometers to meters:
    Rsootsphereinit:=ReactorGrid.DValues[12,InterSectionNr+1]*1.0e-9;

with FlowOut do if IntersectionNr>0 then
begin
    ResTime:=2*L/(Flowin.V+Flowout.V);
    if Fuelin.W>NearlyZero then //Flame in combustion zone
begin //WTotalWater is what has entered UNTIL this reactor
    CompositionWithoutFuel:=MixGasComposition(Waterin.Composition,
        Oxidn.Composition, WTotalWater, WTotalOxid);
    if not StoichFuelFlow(Fuelin.Composition, CompositionWithoutFuel, HCratio,
        StoichFuelFlowPerKgOxid) then
begin //Phi, eq. ratio can't be calculated:
        //use chemical eq. ratio as approximation.
        LocalComposition:=MixGasComposition(Fuelin.Composition,
            CompositionWithoutFuel,
            WTotalFuel, WTotalWater+WTotalOxid);
        EqRatio:=ChemicalEqRatio(LocalComposition, HCratio);
        end
    else //Calculate phi, eq. ratio (fuel/oxid/(fuel/oxid)stoich).
        EqRatio:=(WTotalFuel/(WTotalOxid+WTotalWater))/StoichFuelFlowPerKgOxid;

    if not FPromptNO(Flowin, Fuelin, Oxidn, Waterin, EqRatio,
        Ps, A, PromptNO) then exit;
    if not FFuelNO(Flowin.W, Fuelin.W, Oxidn.W, Waterin.W, molweight,
        FuelNO) then exit;
    if not FCORise(Fuelin.Composition, Composition, Rho, FuelIn.W, W,
        COproduction)
        then exit; //COproduction is in kmol/m3

        //Try smoke production with equivalence ratio for the moment;
        //that enables a more honest comparison between low calorific
        //and other fuels and reduces the situation to air as oxid.
    if not FSmokeProduction(EqRatio, WTotalOxid, Ts, HCratio, Ps,
        Rsootsphereinit, SmokeProd) then exit;
    if not FUHCPProduction(Fuelin.W, Rho, Flowout.W, Fuelin.Composition,
        UHCPProduction) then exit;

    //Conversion from mole fraction to kmol/m3. The amount of promptNO is
    //multiplied with the fraction that the locally inserted fuel is of
    //total fuel inserted sofar. Otherwise very small amounts of added fuel

```



```

//could result in high amounts of prompt NOx, which is not realistic.
NOpandf:=(PromptNO*Fuelin.W/WTotFuel+FuelNO)*Rho/molweight
          +Oxidin.Composition[gtNO]*Oxidin.W/Flowout.W
          *Rho/GasProperties[gtNO].Molemass;
end
else //No flame in combustion zone
begin
NOpandf:=0+Oxidin.Composition[gtNO]*Oxidin.W/Flowout.W
          *Rho/GasProperties[gtNO].Molemass;
//=NO prompt and fuel (=0) and NO coming in from cooling oxid.

COproduction:=0;
SmokeProd:=0;
UHCProduction:=0;
end;

//Calculate NO emission and dNODt at the outlet of the combustion zone
if not FindExitEmissions(etNOx,Gasconc,NOin,dNODtin,NOpandf,ResTime,
                        Ps,NOxTemperature,1e-10,dNODtout, NOout) then Exit;

//Calculate CO and dCODt at the outlet of the combustion zone
if not FindExitEmissions(etCO,Gasconc,COin,dCODtin,COproduction,ResTime,Ps,
                        COTemperature,1e-10, dCODtout, COout) then Exit;

//Compare calculated CO with the local equilibrium CO; if the equilibrium
//amount is higher, the equilibrium values are used, assuming that in that
//case, there was time enough to reach equilibrium.
if (COout<Gasconc[gtCO]) then
begin
COout:=Gasconc[gtCO];
dCODtout:=EmissionFormationRate(etCO, Gasconc, Ps, Ts, COout);
end;

//Calculate SootSphereRadiusout and NumberOfSmokeSpheresout at the outlet
//of the combustion zone. Start with the SootSphereRadiusout:
if not FindExitEmissions(etSmoke, Gascompnewv, SootSphereRadiusin,
                        dSootSphereRadiusdtin, SmokeProd, ResTime, Ps,
                        SmokeTemperature, 1e-10, dSootSphereRadiusdtout,
                        SootSphereRadiusout) then exit;

if (SootSphereRadiusout<NearlyZero) then //No soot left.
    NumberOfSmokeSpheresout:=0
else NumberOfSmokeSpheresout:=NumberOfSmokeSpheresin;
if (SmokeProd>NearlyZero) then //A new NumberOfSmokeSpheresout and
begin //SootsphereRadiusout are calculated.
    SootSphereRadiusout:=NumberOfSmokeSpheresout*SootSphereRadiusout
                        +SmokeProd*Rsootsphereinit
                        /(NumberOfSmokeSpheresout+SmokeProd);
    NumberOfSmokeSpheresout:=NumberOfSmokeSpheresout+SmokeProd;
end;

//Calculate UHC emission and dUHCdt at the outlet of the combustion zone
if not FindExitEmissions(etUHC, Gasconc, UHCin, dUHCdtin, UHCProduction,
                        ResTime, Ps, UHCtemperature, 1e-10, dUHCdtout,
                        UHCout) then exit;

end
else //Intersectionnr = 0: (first) flame zone; fuel must be present,
//otherwise there is no flame; because only the water injected
//UNTIL the specific zone is used to calculate the local equivalence
//ratio, water is not important here.
begin
if not StoichFuelFlow(Fuelin.Composition, Oxidin.Composition, HCRatio,
                    StoichFuelFlowPerKgOxid) then
begin //Phi, eq. ratio can't be calculated:
//use chemical eq ratio as approximation.
LocalComposition:=MixGasComposition(Fuelin.Composition, Oxidin.Composition,
                                    Fuelin.W, Oxidin.W);
EqRatio:=ChemicalEqRatio(LocalComposition, HCRatio);
end
else //Calculate phi, eq. ratio (fuel/oxid/(fuel/oxid)stoich)
EqRatio:=Fuelin.W/Oxidin.W/StoichFuelFlowPerKgOxid;

```

```

if not FPromptNO(Flowin, Fuelin, Oxidin, Waterin, EqRatio,
                 Ps, A, PromptNO) then exit;
if not FFuelNO(Flowin.W, Fuelin.W, Oxidin.W, Waterin.W, molweight,
               FuelNO) then exit;
if not FCORise(Fuelin.Composition, Composition, Rho, FuelIn.W, W,
               COproduction) then exit;
if not FSmokeProduction(EqRatio, WTotalOxid, Ts, HCratio, Ps,
                         Rsootsphereinit, SmokeProd) then exit;
if not FUHCPProduction(Fuelin.W, Rho, Flowout.W,
                       Fuelin.Composition, UHCPProduction) then exit;
with Flowout do
begin
  // new v8.1 1-9-1998 first NOout= Oxidin NO; then add Prompt & Fuel NOx:
  NOout:=Oxidin.Composition[gtNO]*Oxidin.W/Flowout.W
    *Rho/GasProperties[gtNO].Molemass;
  if WtotalFuel>NearlyZero then

  //Convert Prompt and Fuel NOx from molefraction to NO (kmole/m3).
  NOout:=NOout + (PromptNO*Fuelin.W/WTotalFuel+FuelNO)*Rho/molweight;

  dNOdtout:=EmissionFormationRate(etNOx, Gasconc, Ps, NOxTemperature, NOout);
  COout:=COproduction+COin*Rho/molweight;
  dCOdtout:=EmissionFormationRate(etCO, Gasconc, Ps, COTemperature, COout);
  NumberOfSmokeSpheresout:=SmokeProd; //Radius of spheres 40 (nm);
  SootSphereRadiusout:=Rsootsphereinit; //Set soot sphere to initial value
  dSootSphereRadiusdtout:=EmissionFormationRate(etSmoke, gascompnewv,
                                                  Ps, SmokeTemperature, Dummy);
  UHCout := UHCProduction;
  dUHCdtout := EmissionFormationRate(etUHC, gasconc, Ps,
                                     UHCTemperature, UHCout);

end;
end;
end;
Result:=true;
end;

```

FCORise (Combustn.Pas)

```

//This function calculates the CO concentration rise due to combustion in
//kmol/m3 if all the C from the fuel would instantly be converted to CO.
function TCombustor.FCORise(const Fuelin, Flowout: TGasComposition;
                           const Rho, FuelFlow, Wout: Double;
                           var COproduction: Double): Boolean;

var
  GasComp: TGascomposition;
  CinFuel: Double;
begin
  Result:=false;
  GasComp := HCratioMoleComposition(atomweightC+atomweightH*HCratio, FuelComposition);
  CinFuel := GasComp[gtCO2] + GasComp[gtCO] + GasComp[gtCH4] + 2*GasComp[gtC2H6]
    + 2*GasComp[gtC2H4] + 3*GasComp[gtC3H8] + 4*GasComp[gtC4H10]
    + GasComp[gtCxHy];
  //CinFuel is number of moles of C-atoms per mole fuel
  COproduction := CinFuel / FuelGasProps.MoleMass * FuelFlow / Wout * Rho;
  Result:=true;
end;

```

FCp (GSPglobal.Pas)

```

//The aim of this function is to calculate the specific heat at constant
//pressure for a mixture using Cp-values calculated by FCpPerSpecie for the
//different species (equation C.23).
function FCp(const T: Double; const GC:TGasComposition): Double;
var
  i : TGas;
begin
  Result:=0;
  for i:=Low(i) to High(i) do if GC[i]>NearlyZero then

```

```

    Result:=Result+GC[i]*FCpPerSpecie(T, GasProperties[i]);
end;

```

FCpPerSpecie (GSPglobal.Pas)

```

//The purpose of this function is to calculate the specific heat at constant
//pressure (cp) for a given specie at a given temperature (equation C.22).
function FCpPerSpecie(const T: Double; const GP: TGasProperties): Double;
begin
    //CoefsChangeTemp is the bordering temperature between the
    //two sets of (NASA) coefficients (here 1000 (K)).
    with GP do if T>CoefsChangeTemp then
        Result:=(Coefs_above1000[1]*IntPower(T,-2)
            +Coefs_above1000[2]*IntPower(T,-1)
            +Coefs_above1000[3]
            +Coefs_above1000[4]*T
            +Coefs_above1000[5]*IntPower(T, 2)
            +Coefs_above1000[6]*IntPower(T, 3)
            +Coefs_above1000[7]*IntPower(T, 4)
            +Coefs_above1000[8]*IntPower(T, 5)) * (Gasconst/MoleMass)
    else
        Result:=(Coefs_below1000[1]*IntPower(T,-2)
            +Coefs_below1000[2]*IntPower(T,-1)
            +Coefs_below1000[3]
            +Coefs_below1000[4]*T
            +Coefs_below1000[5]*IntPower(T, 2)
            +Coefs_below1000[6]*IntPower(T, 3)
            +Coefs_below1000[7]*IntPower(T, 4)
            +Coefs_below1000[8]*IntPower(T, 5)) * (Gasconst/MoleMass);
    end;
end;

```

FCs (GSPglobal.Pas)

```

//This function is used to find the speed of sound at a temperature in a medium
//using equation 3.2.
function FCs(const T:double;const GasComp:TGasComposition):double;
begin
    Result:=Sqrt(FGm(T, GasComp)*FR(GasComp)*T);
end;

```

FFuelNO (Combusn.Pas)

```

//This function calculates the fuelNOx formed as a mole fraction.
function TCombustor.FFuelNO(const WFlow, WFuel, WOxid, WWater,
    MixtureMoleMass: double;
    var FuelNO: double): boolean;
begin
    //In interface a conversion fraction is given.
    Result:=false;
    FuelNO:=(FBNperc.Value/atomweightN)/100*MixtureMoleMass
        *WFuel/(WFuel+WOxid+WWater+WFlow)*FuelNconvfrac.Value;
    Result:=true;
end;

```

FGm (GSPglobal.Pas)

```

//The aim of this function is to calculate the ratio of specific heats (gamma)
//for a given composition using equation C.10.
function FGm(const T:double;const GasComp:TGasComposition):double;
var
    Cp:double;
begin
    Cp:=FCp(T, GasComp);
    Result:=Cp/(Cp-FR(GasComp));
end;

```

FH (GSPglobal.Pas)

```

//This function calculates the enthalpy of a mixture at a given temperature
//using equation C.26. The function EnthalpyPerSpecie is used to calculate the
//enthalpy of individual species (equation C.25). The absolute value of the

```

```
//enthalpy is determined by the condition that the enthalpy for a specie at
//T = 298.15 (K) is equal to the formation enthalpy of that specie at
//T = 298.15 (K).
function FH(const T: Double; const GC:TGasComposition): Double;
var
  i : TGas;
begin
  Result:=0;
  for i:=Low(i) to High(i) do if GC[i]>NearlyZero then
    Result:=Result+GC[i]*EnthalpyPerSpecie(T, GasProperties[i]);
  end;
```

FindExitEmissions (Combusn.Pas)

```
//The purpose of this function is to find the emissions levels and formation
//rates at each reactor exit using the emission levels and formation rates at
//the reactor entrance by applying the trapezium rule (Crank-Nicolson). The
//function (TCombustor.)'EmissionFormationRate' is used to find the emission
//formation rate as a function of (a.o.) the emission levels.
function TCombustor.FindExitEmissions(const Emission : TEmission;
const Gascomp: TGascomposition;
const Emissionin, dEmissiondtin, ExtraEmission,
ResTime, Ps, Ts, Accuracy: double;
var dEmissiondtout, Emissionout: Double):
Boolean;
var
  i, AFQcode : integer;
  Emissionoutiter, dEmissiondt1, dEmissiondt2, err, dir : Double;
begin
  //Gascomp is gas composition in KMole / m3 ! Emission in kMole / m3
  //ExtraEmission used for prompt formation of emissions (excl. smoke): for NOx
  //the sum of Fuel and Prompt NOx; for CO and UHC the formation in the flame;
  //If smoke is produced, the number of spheres is changed in
  //'EquilibriumReactor'.
  Result:=false;
  if (Emission=etSmoke) then
    begin //For smoke, dEmissiondtout is not depending Emissionout, so the
      //numerical integration is not implicit and no iteration is needed.
      dEmissiondtout:=EmissionFormationRate(Emission, Gascomp, Ps, Ts, Emissionout);
      Emissionout:=ResTime*(dEmissiondtin+dEmissiondtout)/2+Emissionin;
      if Emissionout<0 then Emissionout:=0;
    end
  else
    begin
      i:=0;
      ResetAFQmem;
      Emissionout:=Emissionin; //Initial emissionout guess equal to emissionin !
      repeat
        dEmissiondt1:=EmissionFormationRate(Emission, Gascomp, Ps, Ts, Emissionout);
        //dEmissiondt1 must be equal to dEmissiondt2, calculated using the method
        //of Crank-Nicolson (trapezium rule). Both are a function of Emissionout.
        dEmissiondt2:=2*(Emissionout-Emissionin-ExtraEmission)/ResTime-dEmissiondtin;
        err:=(dEmissiondt1-dEmissiondt2);
        //'err' Must be divided by a constant(!) with the same order of magnitude
        //as dEmissionsdt. Because this constant is not present, a relatively
        //large Accuracy is used here.
        if i=0 then dir:=0.95 else dir:=err;
        AFQcode:=AFQUIR(Emissionout, err, 0, Accuracy, dir, 40,
          (Owner as TGSPcomp).CompIDstr, EmissionoutIter);
        Emissionout:=Emissionoutiter;
        if Emissionout<0 then Emissionout:=0;
        inc(i);
      until (AFQcode>1) or ((i>5) and (Emissionout<NearlyZero));
      if AFQcode=3 then exit; //Error in AFQuir iteration.
      dEmissiondtout:=EmissionFormationRate(Emission, Gascomp, Ps, Ts, Emissionout);
    end;
  Result:=true;
end;
```

FormationEnthalpy (GSPglobal.Pas)

```
//The aim of this function is to calculate the formation enthalpy for a medium.
function FormationEnthalpy(const Composition:TGasComposition) : Double;
var
  ig : TGas;
  Hform : Double;
begin
  Hform:=0;
  for ig:=Low(ig) to High(ig) do
    //Assume formation enthalpy N2, O2, Ar en H2 zero :
    //The formation enthalpy of CxHy is added elsewhere.
    if not (ig in [gtN2, gtO2, gtAr, gtH2, gtCxHy]) then
      if Composition[ig]>NearlyZero then
        Hform:=Hform+Composition[ig]*GasProperties[ig].Hformation
        /GasProperties[ig].Molemass;
    //Note that Molemass unit is grammes, so for Result in J, multiply by 1000.
  Result:=1000*Hform;
end;
```

FormEnthFromHeatValue (GSPglobal.Pas)

```
//The purpose of this function is to convert the user specified heating value
//into a(n artificial) formation enthalpy of H2 or C1H(H/C).
function FormEnthFromHeatValue(const HCratio, HeatingValue: Double): Double;
begin
  if HCratio>1.0e12 then //Assume H2 is the fuel
    Result:=Heatingvalue
    //H2O formation enthalpy J/Mole: (negative value)
    +GasProperties[gtH2Og].Hformation
    //Conversion to J/Kg:
    /GasProperties[gtH2].MoleMass*1000
  else //Assume CxHy hydrocarbon fuel
    Result:=HeatingValue
    +((H2O formation enthalpy J/Mole: (negative value)
    GasProperties[gtH2Og].Hformation*HCratio/2
    //CO2 formation enthalpy J/Mole: (negative value)
    +GasProperties[gtCO2].Hformation)
    //Conversion to J/Kg:
    /(atomweightC+atomweightH*HCratio)*1000;
  end;
```

FPromptNO (Combustn.Pas)

```
//The aim of this function is to calculate the amount of prompt-NOx produced
//in hydrocarbon flames.
function TCombustor.FPromptNO(const Flowin, Fuelin, Oxidin,
                               Waterin: TGasconditions;
                               const PhiEqRatio, P, A: double;
                               var PromptNO: double): boolean;
var PhiFunction, MolHCRatio, StoichNOEq: double;
begin
  Result:=false;
  MolHCRatio:=MolarHCRatio(atomweightC+atomweightH*HCratio,Fuelin.Composition);

  //upper limit of 1.65 only valid if ethylene concentration is small!!
  if (MolHCRatio<NearlyZero) or (PhiEqRatio<=0.6) or (PhiEqRatio>=1.65) then
    PromptNO:=0
  else
    begin //Absolute value is taken because the polynom
          //could become negative around 1.6
          PhiFunction:=abs(0.72706908078726*IntPower(PhiEqRatio,5) //See memorandum,
          -3.86935072432971*IntPower(PhiEqRatio,4) //Figure 6.3.
          +7.85037810631911*IntPower(PhiEqRatio,3)
          -7.60023172369984*IntPower(PhiEqRatio,2)
          +3.55167189367785*PhiEqRatio-0.645780379414646);
    if not StoichNOEqilibrium(Flowin, Oxidin, Waterin, Fuelin, A,
                              StoichNOEq) then
      begin
```



```

        MessageDlg('No convergence in stoichiometric equilibrium NO',mtError,
                    [mbOK], 0);
    exit;
end;
PromptNO:=MolHCRatio*PhiFunction*sqrt(P/100000)*StoichNOEq; //Equation 6.2.
end;
Result:=true;
end;

```

FR (GSPglobal.Pas)

```

//This function calculates the specific gas constant (R) for a given mixture
//(equation H.1).
function FR(const GasComp:TGasComposition): Double;
var
    ig : TGas;
    R : Double;
begin
    R:=0;
    //Liquid water is excluded. In this way the universal gas constant R is
    //divided by the molar mass defined by equation C.29. Because this molar mass
    //is the reciprocal of the number of moles of gaseous species per gram of
    //mixture (n), defined by equation C.28, equation H.1 from the memorandum is
    //found.
    for ig:=Low(ig) to High(ig) do
        if (not (ig in [gtH2O]))
            and (GasComp[ig]>NearlyZero) then
            R:=R+GasComp[ig]/GasProperties[ig].MoleMass;
    Result:=Gasconst*R;
    end;

```

FS (GSPglobal.Pas)

```

//The entropy for a mixture is calculated by this function using equations C.32
//and C.33.
function FS(const T,p:double; const GC:TGasComposition):double;
var
    i
    Result1, Result2, invmolweight, gasfrac
    GCmole
    :
    TGas;
    Double;
    TGasComposition;
begin
    //Because the molweight is needed in this function the molecomposition is
    //calculated here in stead of calling the function molecomposition
    invmolweight:=0;
    Result1:=0;
    for i:=Low(i) to High(i) do if GC[i]>NearlyZero then
        begin
            GCmole[i]:=GC[i]/GasProperties[i].MoleMass;
            invmolweight:=invmolweight+GCmole[i];
        end
    else GCmole[i]:=0;
    for i:=Low(i) to High(i) do GCmole[i]:=GCmole[i]/invmolweight;
    gasfrac:=1-GCmole[gtH2O]; //This is the molar gas fraction of the medium.

    //Result1 uses the function FSperSpecie (equation C.34) to find the Sj0/R for
    //gaseous species (the first right-hand term in equation C.32, which is
    //completely divided by R) subtracts the second right-hand term from equation
    //C.32, and multiplies this with the molefractions for the gaseous part of the
    //medium.
    if T>CoefsChangeTemp then //CoefsChangeTemp is the bordering temperature
        //between the two sets of (NASA) coefficients (here 1000 (K)).
        begin
            for i:=Low(i) to High(i) do with GasProperties[i] do
                if (i>gtH2O) and (GCmole[i]>NearlyZero) then
                    Result1:=Result1+GCmole[i]*
                        (FSperSpecie(T,Coefs_above1000)-ln(GCmole[i]/gasfrac));
            end
        else
            begin

```

```

for i:=Low(i) to High(i) do with GasProperties[i] do
  if (i<>gtH2O1) and (GCmole[i]>NearlyZero) then
    Result1:=Result1+GCmole[i]*
      (FSperSpecie(T,Coefs_below1000)-ln(GCmole[i]/gasfrac));
  end;

//Result2 subtracts the third right-hand term of equation C.32 (pressure
//influence of entropy).
Result2:=Result1-gasfrac*ln(p/100000);

//In the final Result the Sj0/R of condensed species (water) is added and
//after that the S/R for the mixture is multiplied with R.
Result:=(Result2+GCmole[gtH2O1]
  *FSperSpecie(T,GasProperties[gtH2O1].Coefs_below1000))*invmolweight*Gasconst;
end;

```

FSmokeProduction (Combusn.Pas)

```

//The goal of this function is to calculate the smoke production in flames in
//numbers of spheres, using equation 6.29 (=Lefebvre's expression).
function TCombustor.FSmokeProduction(const EqRatio, WTotalOxid, Ts,
                                     HCRatio, Ps3, Rsootsphereinit: double;
                                     var SmokeProd: double): boolean;

const
  SootDensity = 1800;          // (kg/m3)
var
  Hweightcontent, FARatStoich, SootingCHMassFraction : double;

begin
  result:=false;
  case FuelTypeIndex of
    0, 1, 2, 3, 4 : begin          //Jet fuels, diesel or natural gas.
      FARatStoich:=StoichFuelAirRatio(HCRatio, FuelComposition);
      Hweightcontent:=100/(atomweightC/HCRatio+atomweightH);
      if Hweightcontent>18 then Hweightcontent:=18;
      //Otherwise negative smoke productions could occur;
      //no smoke will be produced then.
      SootingCHMassFraction:=1; //Only hydrocarbons present.
    end;
    5 : begin                      //Hydrogen.
      SootingCHMassFraction:=0; //If fuel is H2 no soot is expected
      Hweightcontent:=0; //No hydrocarbons present.
    end;
    6 : begin                      //Fuel with user specified composition.
      //The FARstoich of an average jet fuel is used; only limited smoke
      //production rates are expected for this type of fuel. See
      //discussion in chapter six of memorandum SK.
      FARatStoich:=0.0689;
      SootingCHMassFraction:=FuelComposition[gtCH4]+FuelComposition[gtC2H6]
        +FuelComposition[gtC2H4]+FuelComposition[gtC3H8]
        +FuelComposition[gtC4H10]+FuelComposition[gtCxHy];
      if SootingCHMassFraction<NearlyZero then Hweightcontent:=0
      else Hweightcontent:=(24.93323*FuelComposition[gtCH4]
        +19.95362*FuelComposition[gtC2H6]
        +14.25834*FuelComposition[gtC2H4]
        +18.14199*FuelComposition[gtC3H8]
        +17.204749*FuelComposition[gtC4H10]
        +100/(atomweightC/HCRatio+atomweightH)
        *FuelComposition[gtCxHy])
        //SootingCHMassFraction;
      if Hweightcontent>18 then Hweightcontent:=18; //Otherwise negative smoke
      end; //predicted!
    end; //end case statement
  //This is the modified expression derived from prof Lefebvre's expression:
  //the third line is a correction for the case that not only hydrocarbons are
  //present in fuel; the last line above is conversion from kg to number of
  //spheres.
  SmokeProd:=(0.0145*1e-6*EqRatio*FARatStoich*sqr(Ps3/1000)
    /WTotalOxid/Ts*Power(18-Hweightcontent,1.5))
    *SootingCHMassFraction

```

```
/SootDensity/(4*Pi/3*IntPower(Rsootsphereinit,3));
```

```
result:=true;
end;
```

FSperSpecie (GSPglobal.Pas)

//This function determines the value of S_{j0}/R (the steady state entropy divided //by the universal gas constant, see NASA RP1311, p 74, or the memorandum C.34, //C.32 and C.33) for a specie.

```
function FSperSpecie(const T:double;const GC : TGasCoefArray):double;
begin
```

```
Result:=-GC[1]/2*IntPower(T,-2)
        -GC[2]*IntPower(T,-1)
        +GC[3]*LN(T)
        +GC[4]*T
        +GC[5]/2*IntPower(T,2)
        +GC[6]/3*IntPower(T,3)
        +GC[7]/4*IntPower(T,4)
        +GC[8]/5*IntPower(T,5)
        +GC[10];
```

```
end;
```

FuelFormationandReactantEnthalpy (GSPglobal.Pas)

//This function calculates the fuel formation enthalpy and the fuel reactant //enthalpy per kg for all fuels except fuels with user specified composition. //Also the heating value at the chemical reference temperature (298.15 (K)) is //calculated.

```
function FuelFormationandReactantEnthalpy(const UserSpecCp, HCRatio, THVCp,
                                           Tfuel, LowerHeatingValue: double;
                                           const FuelGasProps : TGasProperties;
                                           var FuelFormationEnthalpyPerKg,
                                           FuelReactantEnthalpyPerKg,
                                           Hv_at_29815 : double): boolean;
```

```
var
    CpdivRoffset, DeltaHReactForHeatValue : double;
begin
    Result:=false;
    FuelFormationEnthalpyPerKg:=0;
    FuelReactantEnthalpyPerKg:=0;
```

```
//Cp/R fuel correction effect factor on Hv: this factor is used to shift the
//Cp(/R) polynomial (thus also the enthalpy polynomial) upwards or downwards.
CpdivRoffset:=(UserSpecCp - FCpperspecie(THVCp, FuelGasProps))
               *FuelGasProps.MoleMass/Gasconst;
```

```
//Hv correction due to heat used for fuel heat up/down from Hv THVCp to
//TrefChemical: necessary to find the heating value at TrefChemical.
DeltaHReactForHeatValue:=EnthalpyChange(THVCp, TrefChemical,FuelGasProps)
                        +GasConst/FuelGasProps.MoleMass*CpdivRoffset*(THVCp-TrefChemical);
```

```
if HCRatio>1.0e12 then //Assume H2 as fuel (HCRatio (probably) = Infinite)
    Hv_at_29815:=LowerHeatingValue //HV_at_298.15=Heating value at TrefChemical.
                +GasProperties[gtH2Og].MoleMass/FuelGasProps.MoleMass
                *EnthalpyChange(THVCp, TrefChemical,GasProperties[gtH2Og])
                -GasProperties[gtO2].MoleMass/FuelGasProps.MoleMass/2
                *EnthalpyChange(THVCp, TrefChemical,GasProperties[gtO2])
                -DeltaHReactForHeatValue
```

```
else //Assume fuel containing only CxHy
    Hv_at_29815:=LowerHeatingValue
                +HCRatio/2*GasProperties[gtH2Og].MoleMass/FuelGasProps.MoleMass
                *EnthalpyChange(THVCp, TrefChemical,GasProperties[gtH2Og])
                +GasProperties[gtCO2].MoleMass/FuelGasProps.MoleMass
                *EnthalpyChange(THVCp, TrefChemical,GasProperties[gtCO2])
                -GasProperties[gtO2].MoleMass/FuelGasProps.MoleMass/2
                *EnthalpyChange(THVCp, TrefChemical,GasProperties[gtO2])
                -DeltaHReactForHeatValue;
```



```

FuelFormationEnthalpyPerKg:=FormEnthFromHeatValue(HCratio,Hv_at_29815);
FuelReactantEnthalpyPerKg:=EnthalpyChange(Tfuel,TrefChemical, FuelGasProps)
+GasConst/FuelGasProps.MoleMass*CpdivRoffset*(TFuel-TrefChemical);
Result:=true;
end;

```

FUHCProduction (Combusn.pas)

```

//This function is used to find the production of unburned hydrocarbons
//in reactors containing a flame. The value of UHCProduction is in (kmole/m3)
function TCombustor.FUHCProduction(const Wfreactor, Rho, Wtotal : double;
const FuelComposition : TGascomposition;
var UHCProduction: double): boolean;

var
  KmoleFuel : double;
begin
  Result := false;
  case FuelTypeIndex of
    0, 1, 2, 3: KmoleFuel := Wfreactor / 167.31462; //Fuel assumed to be C12H23
    4: KmoleFuel := Wfreactor / 16.04276; //Fuel assumed to be CH4
    5: KmoleFuel := 0; //Hydrogen does not produce UHC.
    6: KmoleFuel := Wfreactor*(FuelComposition[gtCH4] + FuelComposition[gtC2H6]
+FuelComposition[gtC2H4] + FuelComposition[gtC3H8]
+FuelComposition[gtC4H10]+ FuelComposition[gtCxHy])
/ 16.04276; //Fuel assumed to be CH4

  end;
  UHCProduction := KmoleFuel * Rho / Wtotal;
  Result:=true;
end;

```

GetStaticfromM (GSPglobal.Pas)

```

//The aim of this function is to calculate the static pressure and temperature,
//as well as the flow speed and the flow cross section area for a given Mach
//number and given total temperature and pressure and composition.
function GetStaticfromM(const M, Pt, Tt, W: double;
const Composition: TGasComposition;
var Ps, Ts, V, A: double): boolean;
: integer;

var
  i
  Gammal, Tsiter : double;
begin
  Result:=false;
{ i:=0;
  Ts:=Tt; //first guess
  ResetAFQmem;
  repeat
    Gammal:=FGm(Ts, Composition);
    Gamma2:=1+2/sqr(M)*(Tt/Ts-1);
    err:=Gammal-Gamma2;
    if i=0 then dir:=0.95
    else dir:=err;
    AFQcode:=AFQUIR(Ts,err,0,0.001,dir,40,
'Combustor in GetStaticFromM iteration',Tsiter);

    Ts:=Tsiter;
    inc(i);
  until AFQcode>1;
  if AFQcode=3 then exit; }
  i:=0;
  Tsiter:=Tt; //First guess
  repeat //Iteration to find proper Ts.
    Ts:=Tsiter;
    Gammal:=FGm(Ts, Composition); //Calculate gamma for guessed Ts.
    Tsiter:=Tt/(1+(Gammal-1)/2*sqr(M)); //Calculate new Ts guess from gamma
    inc(i); //and Mach-number.
  until (abs(Ts-Tsiter)<0.001) or (i>24);
  if i>24 then
  begin
    MessageDlg('No convergence in GetStaticfromM iteration',mtError, [mbOK], 0);
    Exit;
  end;

```

```

end;

//Calculation of other data:
V:=M*FCs(Ts, Composition);
Ps:=Pt/(Power(1+(Gammal-1)/2*sqr(M), (Gammal/(Gammal-1))));
A:=W*V/(Gammal*Ps*sqr(M));
Result:=true;
end;

```

GetStaticfromV (GSPglobal.Pas)

```

//The aim of this function is to calculate the static pressure and temperature,
//as well as the Mach-number and the flow cross section area for a given flow
//speed and given total temperature and pressure and composition.
function GetStaticfromV(const V, Pt, Tt, W: double;
    const Composition: TGasComposition;
    var Ps, Ts, M, A: double): boolean;

var
    i
        : integer;
    Gammal, Tsiter
        : double;
begin
    Result:=false;
    { i:=0;
      Ts:=Tt; //first guess
      ResetAFQmem;

    repeat
        Gammal:=FGm(Ts, Composition);
        M:=V/FCs(Ts, Composition);
        Gamma2:=1+2/sqr(M)*(Tt/Ts-1);
        err:=Gammal-Gamma2;
        if i=0 then dir:=0.95
        else dir:=err;
        AFQcode:=AFQUIR(Ts,err,0,0.001,dir,40,
            'Combustor in GetStaticFromV iteration',Tsiter);

        Ts:=Tsiter;
        inc(i);
    until AFQcode>1;
    if AFQcode=3 then exit;}
    i:=0;
    Tsiter:=Tt; //First guess.
    repeat //Iteration to find proper Ts.
        Ts:=Tsiter;
        Gammal:=FGm(Ts, Composition); //Calculate gamma for guessed Ts.
        M:=V/FCs(Ts, Composition); //Calculate Mach-number for guessed Ts.
        Tsiter:=Tt/(1+(Gammal-1)/2*sqr(M)); //Calculate new value for Ts using
        inc(i); //calculated gamma and Mach-number.
    until (abs(Ts-Tsiter)<0.001) or (i>24);
    if i>24 then
        begin
            MessageDlg('No convergence in GetStaticfromV iteration',mtError, [mbOK], 0);
            Exit;
        end;

    //Calculation of other data.
    Ps:=Pt/(Power(1+(Gammal-1)/2*sqr(M), (Gammal/(Gammal-1))));
    A:=W*V/(Gammal*Ps*sqr(M));
    Result:=true;
end;

```

HCratioMassComposition (GSPglobal.Pas)

```

//This function converts the molecomposition into a masscomposition. A component
//CxHy is allowed to be present in the molecomposition.
function HCratioMassComposition(const CxHyMoleMass: Double;
    const MoleComposition: TGasComposition): TGasComposition;
//The molemass is based on ClH(H/C).

var
    i
        : TGas;

```

```

    molweight    : Double;
    GC           : TGasComposition;
begin
if (abs(MoleComposition[gtCxHy]-1)<NearlyZero) then //Fuel contains only CxHy.
    Result:=MoleComposition
else
begin
    if (abs(MoleComposition[gtCxHy])<NearlyZero) then //User specified fuel com-
    begin                                           //position without CxHy.
        molweight:=0;
        for i:=Low(i) to High(i) do if MoleComposition[i]>NearlyZero then
            begin
                GC[i]:=MoleComposition[i]*GasProperties[i].MoleMass;
                molweight:=molweight+GC[i];
            end
        else GC[i]:=0;
        for i:=Low(i) to High(i) do Result[i]:=GC[i]/molweight;
        end
    else //User specified fuel with CxHy component present.
        begin
            molweight:=0;
            for i:=Low(i) to High(i) do
                if (not (i in [gtCxHy])) and (MoleComposition[i]>NearlyZero) then
                    begin
                        GC[i]:=MoleComposition[i]*GasProperties[i].MoleMass;
                        molweight:=molweight+GC[i];
                    end
                else GC[i]:=0;
                GC[gtCxHy]:=MoleComposition[gtCxHy]*CxHyMoleMass;
                molweight:=molweight+GC[gtCxHy];
                for i:=Low(i) to High(i) do Result[i]:=GC[i]/molweight;
            end;
        end;
    end;
end;
end;

```

HCratioMoleComposition (GSPglobal.Pas)

//This function converts the masscomposition into a molecomposition, even if the
//masscomposition contains a component CxHy.

```

function HCratioMoleComposition(const CxHyMoleMass: Double;
    const MassComposition: TGasComposition):TGasComposition;
    //The CxHyMoleMass is based on ClH(H/C)
var
    i           : TGas;
    invmolweight: Double;
    GC          : TGasComposition;
begin
if (abs(MassComposition[gtCxHy]-1)<NearlyZero) then //Fuel contains only CxHy.
    Result:=MassComposition
else
begin
    if (abs(MassComposition[gtCxHy])<NearlyZero) then //User specified fuel com-
    begin                                           //position without CxHy.
        invmolweight:=0;
        for i:=Low(i) to High(i) do if MassComposition[i]>NearlyZero then
            begin
                GC[i]:=MassComposition[i]/GasProperties[i].MoleMass;
                invmolweight:=invmolweight+GC[i];
            end
        else GC[i]:=0;
        for i:=Low(i) to High(i) do Result[i]:=GC[i]/invmolweight;
        end
    else //User specified fuel with CxHy component present.
        begin
            invmolweight:=0;
            for i:=Low(i) to High(i) do
                if (not (i in [gtCxHy])) and (MassComposition[i]>NearlyZero) then
                    begin
                        GC[i]:=MassComposition[i]/GasProperties[i].MoleMass;
                        invmolweight:=invmolweight+GC[i];
                    end
                else GC[i]:=0;
                GC[gtCxHy]:=MassComposition[gtCxHy]/CxHyMoleMass;
                invmolweight:=invmolweight+GC[gtCxHy];
                for i:=Low(i) to High(i) do Result[i]:=GC[i]/invmolweight;
            end;
        end;
    end;
end;
end;

```

```

        end
        else GC[i]:=0;
        GC[gtCxHy]:=MassComposition[gtCxHy]/CxHyMoleMass;
        invmolweight:=invmolweight+GC[gtCxHy];
        for i:=Low(i) to High(i) do Result[i]:=GC[i]/invmolweight;
        end;
    end;
end;
end;

```

MassComposition (GSPglobal.Pas)

```

//The purpose of this function is to convert a molecomposition into a
//masscomposition.
function MassComposition(const MoleComposition:TGasComposition):TGasComposition;
var
    i          : TGas;
    molweight: double;
    GC         : TGasComposition;
begin
    molweight:=0;
    for i:=Low(i) to High(i) do if MoleComposition[i]>NearlyZero then
        begin
            GC[i]:=MoleComposition[i]*GasProperties[i].MoleMass;
            molweight:=molweight+GC[i];
        end
    else GC[i]:=0;
    for i:=Low(i) to High(i) do Result[i]:=GC[i]/molweight;
    end;
end;

```

MixGasComposition (GSPglobal.Pas)

```

//The purpose of this function is to mix two compositions without reactions
//occurring.
function MixGasComposition(const GC1,GC2 : TGasComposition;
                           const M1,M2   : Double): TGasComposition;
var
    ig : TGas;
begin
    for ig:=Low(ig) to High(ig) do Result[ig]:=(M1*GC1[ig]+M2*GC2[ig])/(M1+M2);
    end;
end;

```

MolarHCRatio (GSPglobal.Pas)

```

//new CxHy, Steven Kluiters
//This function calculates the hydrocarbon molar ratio, i.e. the number of moles
//of hydrocarbons relative to the total number of moles in the mixture.
function MolarHCRatio(const CxHyMoleMass: double;
                      const FuelComp: TGasComposition): Double;
var
    GasComp : TGasComposition; //Procedure can be used for UHC calculation
begin
    //and for Prompt NOx calculation.
    GasComp:=HCRatioMoleComposition(CxHyMoleMass, FuelComp);
    Result := GasComp[gtCH4] + GasComp[gtC2H6] + GasComp[gtC2H4] + GasComp[gtC3H8]+
              GasComp[gtC4H10] + GasComp[gtCxHy];
end;

```

MoleComposition (GSPglobal.Pas)

```

//The aim of this function is to calculate the molecomposition from the
//masscomposition.
function MoleComposition(const MassComposition:TGasComposition):TGasComposition;
var
    i          : TGas;
    invmolweight:double;
    GC         : TGasComposition;
begin
    invmolweight:=0;
    for i:=Low(i) to High(i) do if MassComposition[i]>NearlyZero then
        begin

```

```

    GC[i]:=MassComposition[i]/GasProperties[i].MoleMass;
    invmolweight:=invmolweight+GC[i];
  end
else GC[i]:=0;
for i:=Low(i) to High(i) do Result[i]:=GC[i]/invmolweight;
end;

```

MultiReactorLoop (Combustn.Pas)

```

//This function is called once for every combustion chamber used. Unlike
//'TCombustor.SingleReactorLoop' it doesn't only calculate combustor exit
//conditions, but also emission levels. Therefore, it is only used if the multi-
//reactor emission model is used. 'EquilibriumReactor' is used to find exit
//conditions and emission levels for each reactor separately.
function TCombustor.MultiReactorLoop(const Mode : TCalcMode) : Boolean;
var
  iintersection, irow                                     : integer;
//new 20-9-1998, Steven Kluiters
  NOfraction, Normationfactor,
//end new 20-9-1998
  dNOdtin, NOin, dNOdtout, NOout, NOeq, O2, dCOdtin, COin, dCOdtout, COout,
  {EqRatio, ChemEqRatio, Wwater,} Ain, WOxidFrac, molew_div_rho, COeq,
  SootsphereRadiusin, dSootsphereRadiusdtin, NumberOfSmokeSpheresin,
  dUHCdtin, UHCin, ResTime, SootsphereRadiusout, ParticulateMassLoading,
  dSootsphereRadiusdtout, NumberOfSmokeSpheresout, SmokeNumber,
  dUHCdtout, UHCout, WTotalFuel, WTotalOxid, Wtotalwater, L: double;
  ig                                     : TGas;
  GasCompNewm, GasCompNewv, Gasconc    : TGasComposition;
  Flowin, Oxidin, Fuelin, Waterin      : TGasConditions;
begin
  Result:=false;

  //Set initial flow properties (at start of series of reactors) using data from
  //the component before and from the user-interface:
  Flowin:=CombustionIn;
  Flowin.W:=0;
  Oxidin:=CombustionIn;
  Fuelin.Composition:=Fuelcomposition;
  Fuelin.Tt:=Tfuel.Value;
  Waterin.Composition:=Waterinjcomposition;
  Waterin.Tt:=Twaterinj;
  WtotalOxid:=0;
  WtotalFuel:=0;
  WtotalWater:=0;

  //Initially (before the flame) the carbon monoxide, unburned hydrocarbons and
  //smoke concentrations are made equal to zero: they are assumed to be burnt in
  //a subsequent combustion chamber, without influencing new build-up of
  //emissions. The NO(x) concentration is assumed to be non-zero and is known
  //because it is present in the vector describing the composition of the
  //working medium (in this case 'CombustionIn'). For calculational reasons, the
  //'NOin' is made equal to zero.
  dNOdtin:=0;
  NOin:=0;
  dCOdtin:=0;
  COin:=0;
  NumberofSmokeSpheresIn :=0;
  SootSphereRadiusIn     :=0;
  dSootSphereRadiusdtIn  :=0;
  SmokeNumber            :=0;
  UHCin                 :=0;
  dUHCdtin              :=0;

  //Calculate composition and speed, etc. subsequently for each reactor
  with ReactorGrid do
    begin
      for iintersection:=0 to RowCount-2 do
        begin
          irow:=iintersection+1; //Corresponding row in ReactorGrid
          //Calculate

```

```

if Mode=cmDesign then case Vspecifier.Itemindex of
  0 : Out1.A := DValues[1,irow];
  1 : Out1.Mach := DValues[2,irow];
  2 : Out1.V := DValues[3,irow];
end;

L := DValues[4,irow]; // L ignored for iintersection=0 !!
Fuelin.W := DValues[5,irow]*Wf;
Oxidin.W := DValues[6,irow]*CombustionIn.W;
Waterin.W:= DValues[7,irow]*Wwaterinj;
WtotalOxid:=WtotalOxid+Oxidin.W;
WtotalFuel:=WtotalFuel+Fuelin.W;
if not EquilibriumReactor(Mode,dlHigh, Vspecifier.Itemindex,
  iintersection, Flowin, Fuelin, Oxidin,
  Waterin, Out1, L,
  dNODtin, NOin, dCODtin, COin,
  dUHCdtin, UHCin,
  NumberOfSmokeSpheresin, SootSphereRadiusin,
  dSootSphereRadiusdtin,
  WTotalFuel, WTotalOxid, Wtotalwater,
  dNODtout, NOout, NOeq, dCODtout, COout, COeq, O2,
  dUHCdtout, UHCout,
  NumberOfSmokeSpheresout, SootSphereRadiusout,
  dSootSphereRadiusdtout,
  molew_div_rho, ResTime) then Exit;
WtotalWater:=WtotalWater+Waterin.W;

if Vspecifier.Itemindex<>2 then DValues[3,irow]:=Out1.V;
if Vspecifier.Itemindex<>1 then DValues[2,irow]:=Out1.Mach;
//DValues[2,irow] :=Out1.Mach;
//DValues[3,irow] :=Out1.V;
//Additional output is put in ReactorGrid for test purposes:
DValues[13,irow] :=Out1.Ts;
DValues[14,irow] :=Out1.Ps;
//Conversion from kMole/m3 to ppm (volume) with molew_div_rho:
DValues[15,irow]:=NOout*molew_div_rho*1e6; // ppm
DValues[16,irow]:=dNODtout*molew_div_rho*1e6; // ppm
DValues[17,irow]:=NOeq; // ppm
DValues[18,irow]:=COout*molew_div_rho*1e6; // ppm
DValues[19,irow]:=dCODtout*molew_div_rho*1e6; // ppm
DValues[20,irow]:=COeq; // ppm
DValues[21,irow]:=UHCout*molew_div_rho*1e6; // ppm
DValues[22,irow]:=dUHCdtout*molew_div_rho*1e6; // ppm
DValues[23,irow]:=NumberOfSmokeSpheresout;
DValues[24,irow]:=SootSphereRadiusout*1.0e+9; // [nm]
DValues[25,irow]:=dSootSphereRadiusdtout*1.0e+9; // [nm/s]
DValues[26,irow]:=O2; // ppm
//HCratio of 1 as dummy here since there is no CxHy anymore in Out1
DValues[27,irow]:=ChemicalEqRatio(Out1.Composition,1{HCratio});
DValues[28,irow]:=ResTime;
//The output data for a reactor are used as input data for the next
//reactor:
Flowin:=Out1;
dNODtin:=dNODtout;
NOin:=NOout;
dCODtin:=dCODtout;
COin:=COout;
dUHCdtin:=dUHCdtout;
UHCin:=UHCout;
NumberOfSmokeSpheresin:=NumberOfSmokeSpheresout;
SootSphereRadiusin:=SootSphereRadiusout;
dSootSphereRadiusdtin:=dSootSphereRadiusdtout;
//End calculate
end;

NOexit:=NOout; //combustor exit NO concentration [kMole/m3]=last reactor NO
COexit:=COout; //combustor exit CO concentration [kMole/m3]=last reactor CO
UHCexit:=UHCout; //combustor exit UHC concentration [kMole/m3]=last reactor UHC
//new 20-9-1998, Steven Kluiters:
//First the Density (Rho) at the exit is calculated, using the equilibrium
//composition. This density is used to convert the NOout to (kg NO/kg
//mixture). Subsequently, the equilibrium NO-value is replaced by the
//kinetic NO-value. Because the sum of all species will not be one anymore,

```

```

//a normationfactor is calculated, which is the sum of all species with the
//NO replaced. The new (mass) composition is found by dividing the fractions
//by the normation factor. Errors that are made are (a.o.) the fact that the
//equilibrium composition is used (with the equilibrium amount of NO) to
//calculate density, the fact that the relation C : H : O : N changes
//because of the NO replacement and the fact that the temperature should be
//corrected because of the change in composition. The effects of these
//errors are assumed to be small.
with Out1 do
begin
  Rhoexit:=Ps/FR(Composition)/Ts;
  NOfraction:=NOout/Rhoexit*30.00614;
  Normationfactor:=1-Composition[gtNO]+NOfraction;
  Composition[gtNO]:=NOfraction;
  for ig:=Low(ig) to High(ig) do //Scale fractions down to total one.
    Composition[ig]:=Composition[ig]/Normationfactor;
end;
//end new 20-9-1998
//Smoke Number at exit:

ParticulateMassLoading:=4*Pi/3*IntPower(SootSphereRadiusout,3)*1800*NumberOfSmokeSpheres
out*Rhoexit;
  SmokeNrExit:=372761991989.351*IntPower(ParticulateMassLoading,2)
    +8393954.78459032*ParticulateMassLoading;

end;
Result:=true;
end;

```

PhiIJ (GSPglobal.Pas)

```

//This function calculates the viscosity interaction coefficient for two
//species by applying equation C.43.
function PhiIJ(const MoleMassi, MoleMassj, VisCosi, VisCosj: double): double;
begin
  Result:=0.25*sqr(1+sqr(VisCosi/VisCosj)*Power(MoleMassj/MoleMassi,0.25))
    *sqr(2*MoleMassj/(MoleMassi+MoleMassj));
end;

```

PmaxH2Ovapour (GSPglobal.Pas)

```

//This function gives the maximum H2O vapour pressure at a given temperature.
function PmaxH2Ovapour(const T : Double): Double;
begin
  //Result is in N/m2! The formula is valid until at least 400 (K) (error = 1%).
  //Above that temperature the formula is used as an approximation, which
  //slightly gets worse. Just before the critical temperature the error is
  //about 7%.
  Result:=611*Power(10,(7.5*(T-273.15)/(T-35.85)));
end;

```

ProdfromReact (GSPglobal.Pas)

```

//new 28-7-1998 Steven Kluitters, this function calculates (guesses) a new
//composition after a burning process; O, H, OH, NO, N2O and N2 are supposed
//to participate in the reactions; after the reaction, only (of these species)
//N2 can be present. Liquid water is assumed to evaporate. In case of fuel-rich
//combustion (phi>1), the assumption is made that all C is converted to CO and
//all H to H2; of there is not enough oxygen for this, the function will give
//the result 'false'. In other cases, the amount of oxygen that is left
//(ExcessO, see below), is distributed amongst H2 and CO to form H2O and CO2.
//This calculated composition is not realistic, but it is only a start for the
//procedures calculating the equilibrium compositions and temperatures.
function ProdfromReact(const Wfuel, Woxid, HCRatio : Double;
  const FuelComp, OxidComp: TGasComposition;
  var Gascomp: TGasComposition): boolean;
var
  //complete procedure is changed 5-7-1998 and also 28-7-1998!!
  i : TGas;
  ExcessO, PrelCO, PrelH2, CxHyMoleMass: Double;
  //CxHy is only assumed to exist in the fuel, not in the oxid!

```

```

begin
Result:=false;
for i:=Low(i) to High(i) do Gascomp[i]:=0;

if (abs(FuelComp[gtCxHy]-1)<NearlyZero) then //Fuel containing only CxHy
begin
CxHyMoleMass:=atomweightC+atomweightH*HCRatio;
Gascomp[gtO2]:=Woxid*OxidComp[gtO2]+GasProperties[gtO2].Molemass*
(Woxid*(0.5*OxidComp[gtNO]/GasProperties[gtNO].Molemass
+0.5*OxidComp[gtN2O]/GasProperties[gtN2O].Molemass
+0.5*OxidComp[gtO]/GasProperties[gtO].Molemass
+0.25*OxidComp[gtOH]/GasProperties[gtOH].Molemass
-0.5*OxidComp[gtCO]/GasProperties[gtCO].Molemass
-0.5*OxidComp[gtH2]/GasProperties[gtH2].Molemass
-2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
-3.5*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
-3*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
-5*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
-6.5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
-0.25*OxidComp[gtH]/GasProperties[gtH].Molemass)
-Wfuel*(1+HCRatio/4)/CxHyMoleMass);
if (Gascomp[gtO2]>0) then //There is enough oxygen for complete burning
begin
Gascomp[gtCO2]:=GasProperties[gtCO2].Molemass* //Also assume complete
(Wfuel*1/CxHyMoleMass //burning of O, H and OH.
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+OxidComp[gtCO]/GasProperties[gtCO].Molemass
+OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass));
Gascomp[gtAr]:=Woxid*OxidComp[gtAr];
Gascomp[gtH2Og]:=GasProperties[gtH2Og].Molemass*
(Wfuel*HCRatio/2/CxHyMoleMass
+Woxid*(OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+OxidComp[gtH2]/GasProperties[gtH2].Molemass
+2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*OxidComp[gtH]/GasProperties[gtH].Molemass
+0.5*OxidComp[gtOH]/GasProperties[gtOH].Molemass));
Gascomp[gtN2]:=Woxid*(OxidComp[gtN2]+GasProperties[gtN2].Molemass*
(0.5*OxidComp[gtNO]/GasProperties[gtNO].Molemass
+OxidComp[gtN2O]/GasProperties[gtN2O].Molemass));
end
else
begin //ExcessO is the amount of oxygen that is left if all the fuel
//is converted to CO and H2
ExcessO:=-Wfuel/CxHyMoleMass
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+2*OxidComp[gtO2]/GasProperties[gtO2].Molemass
+OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+OxidComp[gtO]/GasProperties[gtO].Molemass
+OxidComp[gtOH]/GasProperties[gtOH].Molemass
+OxidComp[gtNO]/GasProperties[gtNO].Molemass
+OxidComp[gtN2O]/GasProperties[gtN2O].Molemass
-OxidComp[gtCH4]/GasProperties[gtCH4].Molemass //if the oxides cannot
-2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass //contain hydrocar-
-2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass //bons, these lines
-3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass //can be omitted
-4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass);
if (ExcessO < 0) then Exit //There is not enough oxygen to complete
//conversion to CO and H2: exit function.
else //Calculation can proceed.
begin //PrelCO and PrelH2 are the numbers of kmol/s CO and H2
//formed in case there is just enough oxygen for fuel

```



```

//oxidation to CO and H2 (ExcessO = 0).
PrelCO:=Wfuel/CxHyMoleMass
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+OxidComp[gtCO]/GasProperties[gtCO].Molemass
+OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass);
PrelH2:=Wfuel/CxHyMoleMass
+Woxid*(OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2Ol]/GasProperties[gtH2Ol].Molemass
+OxidComp[gtH2]/GasProperties[gtH2].Molemass
+2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*OxidComp[gtH]/GasProperties[gtH].Molemass
+0.5*OxidComp[gtOH]/GasProperties[gtOH].Molemass);
Gascomp[gtCO]:=GasProperties[gtCO].Molemass
*PrelCO*(1-1/(PrelCO+PrelH2)*ExcessO);
//Can this become negative? In theory, it can not become negative; that
//would mean that there is enough oxygen to form CO2 and H2O, so that we
//shouldn't have arrived in this part of the function in the first place.
Gascomp[gtH2]:=GasProperties[gtH2].Molemass
*PrelH2*(1-1/(PrelH2+PrelCO)*ExcessO);
Gascomp[gtCO2]:=GasProperties[gtCO2].Molemass
*(1/(1+PrelH2/PrelCO))*ExcessO;
Gascomp[gtAr]:=Wfuel*FuelComp[gtAr]+Woxid*OxidComp[gtAr];
Gascomp[gtO2]:=0;
Gascomp[gtH2Og]:=GasProperties[gtH2Og].Molemass
*(1/(1+PrelCO/PrelH2))*ExcessO;
Gascomp[gtN2]:=Woxid*(OxidComp[gtN2]+GasProperties[gtN2].Molemass*
(0.5*OxidComp[gtNO]/GasProperties[gtNO].Molemass
+OxidComp[gtN2O]/GasProperties[gtN2O].Molemass));
end;
end;
end
else //Fuel with other components than CxHy;
begin //CxHyMoleMass belongs to component CxHy.
CxHyMoleMass:=atomweightC+atomweightH*HCRatio;
Gascomp[gtO2]:=Wfuel*FuelComp[gtO2]+Woxid*OxidComp[gtO2]
+GasProperties[gtO2].Molemass*
(Wfuel*(0.5*FuelComp[gtNO]/GasProperties[gtNO].Molemass
+0.5*FuelComp[gtN2O]/GasProperties[gtN2O].Molemass
+0.5*FuelComp[gtO]/GasProperties[gtO].Molemass
+0.25*FuelComp[gtOH]/GasProperties[gtOH].Molemass
-0.5*FuelComp[gtCO]/GasProperties[gtCO].Molemass
-0.5*FuelComp[gtH2]/GasProperties[gtH2].Molemass
-2*FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
-3.5*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
-3*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
-5*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
-6.5*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
-0.25*FuelComp[gtH]/GasProperties[gtH].Molemass
-(1+HCRatio/4)*FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(0.5*OxidComp[gtNO]/GasProperties[gtNO].Molemass
+0.5*OxidComp[gtN2O]/GasProperties[gtN2O].Molemass
+0.5*OxidComp[gtO]/GasProperties[gtO].Molemass
+0.25*OxidComp[gtOH]/GasProperties[gtOH].Molemass
-0.5*OxidComp[gtCO]/GasProperties[gtCO].Molemass
-0.5*OxidComp[gtH2]/GasProperties[gtH2].Molemass
-2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
-3.5*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
-3*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
-5*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
-6.5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
-0.25*OxidComp[gtH]/GasProperties[gtH].Molemass));
if (Gascomp[gtO2]>0) then //There is enough oxygen for complete burning

```

```

begin
//also assume complete burning of O, H and OH
Gascomp[gtCO2]:=GasProperties[gtCO2].Molemass*
(Wfuel*(FuelComp[gtCO2]/GasProperties[gtCO2].Molemass
+FuelComp[gtCO]/GasProperties[gtCO].Molemass
+FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+OxidComp[gtCO]/GasProperties[gtCO].Molemass
+OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass));
Gascomp[gtAr]:=Wfuel*FuelComp[gtAr]+Woxid*OxidComp[gtAr];
Gascomp[gtH2Og]:=GasProperties[gtH2Og].Molemass*
(Wfuel*(FuelComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+FuelComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+FuelComp[gtH2]/GasProperties[gtH2].Molemass
+2*FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*FuelComp[gtH]/GasProperties[gtH].Molemass
+0.5*FuelComp[gtOH]/GasProperties[gtOH].Molemass
+HCRatio/2*FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+OxidComp[gtH2]/GasProperties[gtH2].Molemass
+2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*OxidComp[gtH]/GasProperties[gtH].Molemass
+0.5*OxidComp[gtOH]/GasProperties[gtOH].Molemass));
Gascomp[gtN2]:=Wfuel*FuelComp[gtN2]+Woxid*OxidComp[gtN2]+
GasProperties[gtN2].Molemass*
(Wfuel*(0.5*FuelComp[gtNO]/GasProperties[gtNO].Molemass
+FuelComp[gtN2O]/GasProperties[gtN2O].Molemass)
+Woxid*(0.5*OxidComp[gtNO]/GasProperties[gtNO].Molemass
+OxidComp[gtN2O]/GasProperties[gtN2O].Molemass));
end
else //There is not enough oxygen for complete combustion.
begin //ExcessO is the amount of oxygen that is left if all the fuel
//is converted to CO and H2.
ExcessO:=Wfuel*(FuelComp[gtCO2]/GasProperties[gtCO2].Molemass
+2*FuelComp[gtO2]/GasProperties[gtO2].Molemass
+FuelComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+FuelComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+FuelComp[gtO]/GasProperties[gtO].Molemass
+FuelComp[gtOH]/GasProperties[gtOH].Molemass
+FuelComp[gtNO]/GasProperties[gtNO].Molemass
+FuelComp[gtN2O]/GasProperties[gtN2O].Molemass
-FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
-2*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
-2*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
-3*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
-4*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
-FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+2*OxidComp[gtO2]/GasProperties[gtO2].Molemass
+OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+OxidComp[gtO]/GasProperties[gtO].Molemass
+OxidComp[gtOH]/GasProperties[gtOH].Molemass

```

```

+OxidComp[gtNO]/GasProperties[gtNO].Molemass
+OxidComp[gtN2O]/GasProperties[gtN2O].Molemass
-OxidComp[gtCH4]/GasProperties[gtCH4].Molemass //if the oxides cannot
-2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass //contain hydrocar-
-2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass //bons, these lines
-3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass //can be omitted
-4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass);
if (ExcessO < 0) then //There is not enough oxygen to complete
//conversion to CO and H2: exit procedure.
{
begin
MessageDlg('Equivalence ratio too high',mtError,[mbOK],0);}
Exit{; //MessageDlg not needed: the user doesn't have to
end} //know what method of guessing is used.
else //Calculation can proceed
begin //PrelCO and PrelH2 are the numbers of kmol/s CO and H2
//formed in case there is just enough oxygen for fuel
//oxidation to CO and H2 (ExcessO = 0).
PrelCO:=Wfuel*(FuelComp[gtCO2]/GasProperties[gtCO2].Molemass
+FuelComp[gtCO]/GasProperties[gtCO].Molemass
+FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(OxidComp[gtCO2]/GasProperties[gtCO2].Molemass
+OxidComp[gtCO]/GasProperties[gtCO].Molemass
+OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+2*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+3*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+4*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass);
PrelH2:=Wfuel*(FuelComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+FuelComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+FuelComp[gtH2]/GasProperties[gtH2].Molemass
+2*FuelComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*FuelComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*FuelComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*FuelComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*FuelComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*FuelComp[gtH]/GasProperties[gtH].Molemass
+0.5*FuelComp[gtOH]/GasProperties[gtOH].Molemass
+HCRatio/2*FuelComp[gtCxHy]/CxHyMoleMass)
+Woxid*(OxidComp[gtH2Og]/GasProperties[gtH2Og].Molemass
+OxidComp[gtH2O1]/GasProperties[gtH2O1].Molemass
+OxidComp[gtH2]/GasProperties[gtH2].Molemass
+2*OxidComp[gtCH4]/GasProperties[gtCH4].Molemass
+3*OxidComp[gtC2H6]/GasProperties[gtC2H6].Molemass
+2*OxidComp[gtC2H4]/GasProperties[gtC2H4].Molemass
+4*OxidComp[gtC3H8]/GasProperties[gtC3H8].Molemass
+5*OxidComp[gtC4H10]/GasProperties[gtC4H10].Molemass
+0.5*OxidComp[gtH]/GasProperties[gtH].Molemass
+0.5*OxidComp[gtOH]/GasProperties[gtOH].Molemass);

Gascomp[gtCO]:=GasProperties[gtCO].Molemass
*PrelCO*(1-1/(PrelCO+PrelH2)*ExcessO);
//Can this become negative? In theory, it can not become negative; that
//would mean that there is enough oxygen to form CO2 and H2O, so that we
//shouldn't have arrived in this part of the function in the first place.
Gascomp[gtH2]:=GasProperties[gtH2].Molemass
*PrelH2*(1-1/(PrelH2+PrelCO)*ExcessO);
Gascomp[gtCO2]:=GasProperties[gtCO2].Molemass
*(1/(1+PrelH2/PrelCO))*ExcessO;
Gascomp[gtAr]:=Wfuel*FuelComp[gtAr]+Woxid*OxidComp[gtAr];
Gascomp[gtO2]:=0;
Gascomp[gtH2Og]:=GasProperties[gtH2Og].Molemass
*(1/(1+PrelCO/PrelH2))*ExcessO;
Gascomp[gtN2]:=Wfuel*FuelComp[gtN2]+Woxid*OxidComp[gtN2]+
GasProperties[gtN2].Molemass*
(Wfuel*(0.5*FuelComp[gtNO]/GasProperties[gtNO].Molemass
+FuelComp[gtN2O]/GasProperties[gtN2O].Molemass)

```

```

+Woxid*(0.5*OxidComp[gtN0]/GasProperties[gtN0].Molemass
+OxidComp[gtN20]/GasProperties[gtN20].Molemass));
end;
end;
end;

//Conversion of specie mass flows to fractions.
for i:=Low(i) to High(i) do Gascomp[i]:=Gascomp[i]/(Wfuel+Woxid);
Result:=true;
end;

```

ReactantEnthalpy (GSPglobal.Pas)

//The purpose of this function is to calculate the reactant enthalpy, i.e. the //enthalpy change involved in a temperature change between a temperature and the //chemical reference temperature (T = 298.15 (K)) of a fuel flow and oxid flow.

```

function ReactantEnthalpy(const Wfuel,Tfuel,Woxid,Toxid:Double;
const FuelComp,OxidComp:TGasComposition): Double;
begin
Result:=Wfuel*RH(FuelComp,Tfuel)+Woxid*RH(OxidComp,Toxid);
end;

```

ReynoldsNumberIndex (GSPglobal.Pas)

//The purpose of this function is to calculate the Reynolds Number Index, using //equation I.9. The necessary dynamic viscosity is calculated using equation //C.41 and the functions ViscosityPerSpecie and PhiIJ.

```

function ReynoldsNumberIndex(const P, T:double; const MassComp: TGasComposition):
double;
var Mu, PhiSummation      : double;
ig, i                     : TGas;
MoleComp, VisCosArray: TGasComposition;
//Watch out: VisCosArray is not really a composition, but an
//array with the viscosities calculated for each specie present.

begin //Calculate molecomposition and make correction for liquid species.
MoleComp:=MoleComposition(MassComp);
if MoleComp[gtH2O1]>NearlyZero then //Liquid species not included in
for ig:=Low(ig) to High(ig) do //calculation of viscosity.
MoleComp[ig]:=MoleComp[ig]/(1-MoleComp[gtH2O1]);

//Calculate the viscosity per specie for the species present.
for ig:=Low(ig) to High(ig) do with GasProperties[ig] do
begin
if MoleComp[ig]<NearlyZero then VisCosArray[ig]:=0
else //1000 is CoefsChangeTemp
if (T<1000) then VisCosArray[ig] := ViscosityPerSpecie(T, VisCoefs_below1000)
else VisCosArray[ig] := ViscosityPerSpecie(T, VisCoefs_above1000);
end;

//Calculate the viscosity for the mixture (=Mu) using the mole-
Mu:=0; //composition and the viscosity for the individual species.
for ig:=Low(ig) to High(ig) do
if MoleComp[ig]>NearlyZero then
begin
PhiSummation:=0;
for i:=Low(i) to High(i) do
//Calculate the interaction effect of the species on the viscosity.
if not (i = ig) and (MoleComp[i]>NearlyZero) then
PhiSummation:=PhiSummation+MoleComp[i]*PhiIJ(GasProperties[ig].Molemass,
GasProperties[i].Molemass,
VisCosArray[ig], VisCosArray[i]);
Mu:=Mu+MoleComp[ig]*VisCosArray[ig]*1e-7/(MoleComp[ig] + PhiSummation);
end; //1e-7 is a correction to go from micropoise to kg/(ms).

Result:=P/P_slst*Mu/Mu_slst/(T/T_slst*FR(MassComp)/R_st);
end;

```

RH (GSPglobal.Pas)

```
//The aim of this function is to calculate the enthalpy change involved in a
//temperature change from a given temperature to the chemical reference
//temperature of T = 298.15 (K) of a given composition.
function RH(const GC : TGasComposition; const T : Double): Double;
var
  i : TGas;
begin
  Result:=0;
  //For the component CxHy enthalpy changes are calculated and added elsewhere.
  for i:=Low(i) to High(i) do if (i <> gtCxHy) and (GC[i]>NearlyZero) then
    Result:=Result+GC[i]*EnthalpyChange(T,TrefChemical,GasProperties[i]);
  end;
```

SetEquilibriumConditions (GSPglobal.Pas)

```
//This function determines a new temperature and composition in case of
//evaporation (/condensation) and/or dissociation.
function SetEquilibriumConditions(const InfoStr: String;
                                var GasEQ : TGasConditions) : Boolean;

const tempschatting = 1600;
var
  availenth, neededenth, Hform, Hreact, Titer,Titer2, dir, err: double;
  Gas0 : TGasConditions;
begin
  Result:=false;
  Gas0:=GasEQ;
  Titer:=Gas0.Tt;
  // Iteration
  // H2O liquid - vapour equilibrium:
  if (Gas0.Composition[gtH2Ol]>NearlyZero)
    or ((Gas0.Composition[gtH2Og]>NearlyZero) and (Titer<TcritH2O)) then
    begin
      //Check whether equilibrium is correct.
      ResetAFQmem;
      repeat
        GasEQ.Composition:=EquilibriumH2O(Titer,Gas0.Pt,Gas0.Composition);
        //Check if water has changed phase
        if (AFQmem.iter=0)
          and (abs(GasEQ.Composition[gtH2Og]-Gas0.Composition[gtH2Og])<NearlyZero) then
          AFQcode:=2 //Exit, no change in phases, T = Titer.
        else //Calculate new temperature taking into account evaporation
          begin //or condensation.
            Hform:=FormationEnthalpy(Gas0.Composition)
              -FormationEnthalpy(GasEQ.Composition);
            Hreact:=RH(Gas0.Composition,Gas0.Tt);
            availenth:=Hreact+Hform;
            neededenth:=RH(GasEQ.Composition,Titer);
            err:=(availenth-neededenth)/abs(availenth);
            dir:=err;
            AFQcode:=AFQUIR(Titer,err,0,0.001,dir,40,InfoStr+' (H2O)',Titer2);
            Titer:=Titer2;
            if Titer<200 then Titer:=220;
          end;
        until AFQcode>1;
        if AFQcode=3 then Exit; //Error in AFQUIR iteration
      end;
      GasEQ.Wgas:=GasEQ.W*(1-GasEQ.Composition[gtH2Ol]);
      GasEQ.Wgasc:=GasEQ.Wc*(1-GasEQ.Composition[gtH2Ol]);

      Gas0:=GasEQ;
      Titer:=Gas0.Tt;
      //Dissociation effects (towards CO, H2, O2 at T > Tfirst_dissociation 1800 K).
      if (Titer>Tfirst_dissociation)
        // new 16-9-1998
        or (Gas0.Composition[gtOH]>NearlyZero) //Use Dissociation to remove gtOH
        or (Gas0.Composition[gtO]>NearlyZero) //Use Dissociation to remove gtO
        or (Gas0.Composition[gtH]>NearlyZero) //Use Dissociation to remove gtH
      then
        begin
```

```

ResetAFQmem;
repeat
  if not Dissociation(Gas0) then Exit;
  //Only changes Composition at Pt, Tt.
  //Check if dissociation level has changed.
  if (AFQmem.iter=0)
    and (abs(GasEQ.Composition[gtO2]-Gas0.Composition[gtO2])<NearlyZero) then
    AFQcode:=2 //Exit, no change in O2 concentration, T = Titer.
  else //Calculate new temperature taking into account dissociation
    begin //or recombination.
      Hform:=FormationEnthalpy(Gas0.Composition)
      -FormationEnthalpy(GasEQ.Composition);
      Hreact:=RH(Gas0.Composition,Gas0.Tt);
      availenth:=Hreact+Hform;
      neededenth:=RH(GasEQ.Composition,Titer);
      err:=(availenth-neededenth)/abs(availenth);
      dir:=err;
      AFQcode:=AFQUIR(Titer,err,0,0.001,dir,40,InfoStr+' (dissociation)',Titer2);
      Titer:=Titer2;
      if Titer<200 then Titer:=220;
    end;
until AFQcode>1;
if AFQcode=3 then Exit; //Error in AFQUIR iteration
end;
//Else (Titer<=Tfirst_dissociation) then:
//no change in composition; CO and other products form (previous)
//dissociation are FROZEN !!!!

GasEQ.Tt:=Titer;
with GasEQ do H:=FH(Tt,Composition);
Result:=true;
end;

```

SingleReactorLoop (Combustn.Pas)

```

//This function is similar to 'TCombustor.MultiReactorLoop', but here the
//combustion chamber is modelled as one reactor. Also, no emissions are
//calculated. This function is used to find the combustor exit conditions,
//assuming that the NOx concentration at the exit is close to the equilibrium
//value. It is used once for every combustion chamber.
function TCombustor.SingleReactorLoop(const Mode : TCalcMode) : Boolean;
var
  Vspec : integer;
  Dummy : Double;
  ig : TGas;
  GasCompNewm, GasCompNewv, Gasconc : TGasComposition;
  Flowin,Oxidin,Fuelin,Waterin : TGasConditions;
begin
  Result:=false;

  // initial Flow at start of series of reactors:
  Flowin:=CombustionIn;
  Flowin.W:=0;
  Oxidin:=CombustionIn; //Compressor air enters via Oxidin.
  Fuelin.Composition:=Fuelcomposition;
  Fuelin.Tt:=Tfuel.Value;
  Fuelin.W := Wf;
  Waterin.Composition:=Waterinjcomposition;
  Waterin.Tt:=Twaterinj;
  Waterin.W:= Wwaterinj;
  with ReactorGrid do //Calculate equilibrium composition and speed, etc. at
    begin //exit of single reactor with data of last intersection.
      if EmissionModelTypeRGrp.ItemIndex=3 then // multizone model with V specifiers
        begin
          Vspec:=Vspecifier.ItemIndex;
          if Mode=cmDesign then case Vspec of
            0 : Out1.A := DValues[1,RowCount-1];
            1 : Out1.Mach := DValues[2,RowCount-1];
            2 : Out1.V := DValues[3,RowCount-1];
          end;
        end;
      end;
    end;
  end;

```



```

    end
else //In case the new combustor emission model is not used, the Mach number
    //at the combustor exit for calculation of static gas properties is
    //estimated 0.1.
    begin
        Vspec:=1;
        Out1.Mach := 0.1;
    end;
if not EquilibriumReactor(Mode,dllow,Vspec, // specify estimated Mach in reactor
exit
    RowCount-2, Flowin, Fuelin, Oxidin,
    Waterin, Out1, 0,0,0,
    //L, dNodtin, NOin, dNodtout, NOout, etc. are not used, so a dummy 0 is
    //entered here.
    Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,
    Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,Dummy,
    Dummy,Dummy,Dummy,Dummy) then Exit;

if Vspecifier.Itemindex<>2 then DValues[3,RowCount-1]:=Out1.V;
if Vspecifier.Itemindex<>1 then DValues[2,RowCount-1]:=Out1.Mach;

//Additional output to be calculated.
//    DValues[2,RowCount-1] :=Out1.Mach;
//    DValues[3,RowCount-1] :=Out1.V;
DValues[13,RowCount-1] :=Out1.Ts;
DValues[14,RowCount-1] :=Out1.Ps;
{HCratio of 1 as dummy here since there is no CxHy anymore in Out1}
DValues[21,RowCount-1]:=ChemicalEqRatio(Out1.Composition,1{HCratio});
end;
with Out1 do Rhoexit:=Ps/FR(Composition)/Ts;
Result:=true;
end;

```

StoichFuelAirRatio (GSPglobal.Pas)

```

//The aim of this function is to calculate the stoichiometric fuel-air-ratio
//(E.R. = 1) for a given fuel composition
function StoichFuelAirRatio(const HCratio : Double;
    const FuelComp: TGasComposition): Double;
begin
    Result:=1/(GasProperties[gtO2].MoleMass/0.2314151*
        (0.5*FuelComp[gtCO]/GasProperties[gtCO].MoleMass
        -FuelComp[gtO2]/GasProperties[gtO2].MoleMass
        +0.5*FuelComp[gtH2]/GasProperties[gtH2].MoleMass
        +2*FuelComp[gtCH4]/GasProperties[gtCH4].MoleMass
        +3.5*FuelComp[gtC2H6]/GasProperties[gtC2H6].MoleMass
        +3*FuelComp[gtC2H4]/GasProperties[gtC2H4].MoleMass
        +5*FuelComp[gtC3H8]/GasProperties[gtC3H8].MoleMass
        +6.5*FuelComp[gtC4H10]/GasProperties[gtC4H10].MoleMass
        -0.5*FuelComp[gtO]/GasProperties[gtO].MoleMass
        +0.25*FuelComp[gtH]/GasProperties[gtH].MoleMass
        -0.25*FuelComp[gtOH]/GasProperties[gtOH].MoleMass
        -0.5*FuelComp[gtNO]/GasProperties[gtNO].MoleMass
        -0.5*FuelComp[gtN2O]/GasProperties[gtN2O].Molemass
        +(1+HCratio/4)*FuelComp[gtCxHy]/(atomweightC+atomweightH*HCratio)));
    //last two lines can be omitted when NO and N2O are omitted from
    //the user-spec fuel composition interface.
end;

```

StoichFuelFlow (Combustn.Pas)

```

//The aim of this function is to find the fuel flow to be added per kg of oxid
//flow to make the mixture stoichiometric.
function TCombustor.StoichFuelFlow(const FuelComp, OxidComp: TGascomposition;
    const HCratio: Double;
    var StoichFuelFlowPerKgOxid: Double): Boolean;
var
    err, dir, Iterfuelflow, Fuelflownew, AFQcode: Double;
    ig                                     : TGas;

```

```

    StartGasComp                                     : TGascomposition;

begin
Result:=false;
IterFuelFlow:=0.05;
ResetAFQmem;

//Prompt NOx formation cannot be predicted if the flow is already fuel-rich
//without the fuel added.
if ChemicalEqRatio(OxidComp, 0)>1 then
begin
    MessageDlg('Warning: Equivalence ratio of combustion oxidant higher than one',
               mtError, [mbOK], 0);
    exit;
end;

repeat //Search for FuelFlow that makes the combustion stoichiometric.
    for ig:=Low(ig) to High(ig) do //Only one type of fuel may be used.
        StartGasComp[ig]:=( 1*OxidComp[ig]
                             +IterFuelFlow*FuelComp[ig])
                             /(1+IterFuelFlow);

        //Correct value for fuelflownew is found when the (chemical) equivalence
        //ratio is equal to one (err = 0).
        err:=ChemicalEqRatio(StartGasComp, HCratio)-1;
        dir:=err;
        AFQcode:=AFQUIR(IterFuelFlow,err,0,0.001,dir,40,
                        (Self.Owner as TGSPcomp).CompIDstr,FuelFlowNew);
        IterFuelFlow:=FuelFlowNew;
        if IterFuelFlow<=0 then IterFuelFlow:=NearlyZero;
    until AFQcode>1;
    if AFQcode=3 then Exit; //Error in AFQUIR iteration
    StoichFuelFlowPerKgOxid:=IterFuelFlow;
    Result:=true;
end;

```

StoichNOEquilibrium (Combustn.Pas)

```

//This function calculates the stoichiometric equilibrium NO molefraction
//of a given fuel inserted anywhere in the combustion chamber as long as the
//flow from the previous reactor mixed with the oxid entering this reactor is
//not fuel-rich.
function TCombustor.StoichNOEquilibrium(const Flowin, Oxidin, Waterin,
                                       Fuelin: TGasconditions;
                                       const A: double;
                                       var StoichNOEq: double): boolean;

label iteration;
var
    ChemEqRatio, IterFuelFlow, FuelFlowNew, FormEnthalpyWithoutFlowout,
    Hformation, Havail, Hreactant, Hneeded,
    err, dir, AFQcode, Titer, CxHyMoleMass           : Double;
    StartGasComp, Gascompnewv                        : TGasComposition;
    Flowout, InMix                                    : TGasConditions;
    ig                                                 : TGas;
    i                                                 : Integer;
    ConvergedAllowed                                  : Boolean;

begin
Result:=false;
IterFuelFlow:=1;
ResetAFQmem;
for ig:=Low(ig) to High(ig) do
InMix.Composition[ig]:=( Flowin.W*Flowin.Composition[ig]
                        +Oxidin.W*Oxidin.Composition[ig]
                        +Waterin.W*Waterin.Composition[ig] )
                        /(Flowin.W+Oxidin.W+Waterin.W);
InMix.W:=Flowin.W+Oxidin.W+Waterin.W;
if ChemicalEqRatio(InMix.Composition, HCratio)>1 then
    //Prompt NOx formation cannot be predicted if the flow
    //is already fuel-rich without the fuel added

```



```

begin
  MessageDlg('Warning: Equivalence ratio too high for accurate Prompt-NOx
             prediction', mtError, [mbOK], 0);
  StoichNOEq:=0;
  Result:=true;      //Although the promptNOx can't be predicted well, the
  exit;              //calculations proceed.
end;

repeat      //Search for FuelFlow that makes the combustion stoichiometric.
  for ig:=Low(ig) to High(ig) do      //Only one type of fuel may be used.
    StartGasComp[ig]:=( Inmix.W*Inmix.Composition[ig]
                        +IterFuelFlow*Fuelin.Composition[ig])
                        /(Inmix.W+IterFuelFlow);
    //ChemicalEqRatio(StartGasComp, HCRatio, ChemEqRatio);
    err:=ChemicalEqRatio(StartGasComp, HCRatio)-1;
    dir:=err;
    AFQcode:=AFQUIR(IterFuelFlow,err,0,0.001,dir,40,' ',FuelFlowNew);
    IterFuelFlow:=FuelFlowNew;
    if IterFuelFlow<0 then IterFuelFlow:=NearlyZero;
  until AFQcode>1;
  if AFQcode=3 then
    begin
      MessageDlg('No convergence in Afquir IterFuelFlow',mtError, [mbOK], 0);
      Exit; //Error in AFQUIR iteration
    end;

    //Calculate the Equilibrium Composition
    i:=0;
    ResetAFQmem;
    Flowout.Pt:=Flowin.Pt;
    Flowout.A:=A;

    //First a guess of the composition is made by application of ProdFromReact.
    //Using this new composition, a (total) temperature guess is produced.
    if not ProdFromReact(IterFuelFlow, Inmix.W, HCRatio, Fuelin.Composition,
                        Inmix.Composition, Flowout.Composition) then
      begin
        Flowout.Tt:=2000;      //ProdfromReact doesn't work: use rough estimate.
        goto iteration;
      end;

      Flowout.W:=Inmix.W+IterFuelFlow;
      Hreactant:= Flowin.W*RH(Flowin.Composition,Flowin.Tt)
                +Oxidin.W*RH(Oxidin.Composition,Oxidin.Tt)
                +Waterin.W*RH(Waterin.Composition,Waterin.Tt)
                +IterFuelFlow*FuelReactantEnthalpyPerKg;
      FormEnthalpyWithoutFlowout:=Flowin.W*FormationEnthalpy(Flowin.Composition)
                +IterFuelFlow*FuelFormationEnthalpyPerKg
                +Oxidin.W*FormationEnthalpy(Oxidin.Composition)
                +Waterin.W*FormationEnthalpy(Waterin.Composition);
      Hformation:=FormEnthalpyWithoutFlowout
                -Flowout.W*FormationEnthalpy(Flowout.Composition);
      Flowout.Tt:=2000;
      repeat      //Find temperature guess.
        inc(i);
        Titer:=Flowout.Tt;
        Flowout.Tt:=Flowin.Tt+Hformation/Flowout.W/FCp(Titer,Flowout.Composition);
      until (abs(Titer-Flowout.Tt)<40) or (i>16);
      if i>16 then
        begin
          MessageDlg('Reactor exit temp. estimation iteration not converging in'#{+13+
                    (Self.Owner as TGSPComp).CompIDstr}, mtError, [mbOK], 0);
          Exit;
        end;
      end;
    {end
  {end
}else
  begin
    Flowout.W:=Flowin.W+Oxidin.W+Waterin.W+IterFuelFlow;
    Flowout.Tt:=Flowin.Tt;
    Flowout.Composition:=Flowin.Composition;

```



```

end;}

//Start iteration to find correct equilibrium temperature and composition.
iteration:
ResetAFQmem;
i:=0;
repeat
ConvergedAllowed:=true;

//The old composition is used, so there should be an extra
//'repeat-until'-loop here, but it is assumed that in the last loops used to
//find the temperature, the composition is almost constant
with Flowout do if (i>0) then //In the first loop the composition is calcu-
//lated using total temperature and pressure.
begin
if not GetStatic(W, Pt, Tt, A, Composition, Ps, Ts, Mach, V,
'Combustion reactor') then Exit;
if not CombEquilibrium(Ts, Ps, HCRatio, StartGasComp, Gascompnewv,
Composition) then
begin
//old if not ProdFromReact(1, 0, CxHyMoleMass, StartGasComp, Air,
//Composition) then
//new 20-10-1998
if not ProdFromReact(1, 0, HCRatio, StartGasComp, Air, Composition) then
begin
MessageDlg('No convergence in Equilibrium Reactor', mtError, [mbOK], 0);
Exit;
end;
Gascompnewv:=MoleComposition(Composition);
ConvergedAllowed:=false;
end;
end
else //i=0: first iteration loop (total temperature and pressure used).
begin
ConvergedAllowed:=false;
if not CombEquilibrium(Tt, Pt, HCRatio, StartGasComp, Gascompnewv,
Composition) then
begin
//old ProdFromReact(1, 0, CxHyMoleMass, StartGasComp, Air, Composition);
//new 20-10-1998
ProdFromReact(1, 0, HCRatio, StartGasComp, Air, Composition);
Gascompnewv:=MoleComposition(Composition);
end;
end;

Hformation:=FormEnthalpyWithoutFlowout
-Flowout.W*FormationEnthalpy(Flowout.Composition);
Havail:=Hreactant+Hformation; //Eventually apply heat loss here
with Flowout do Hneeded:=W*RH(Composition, Tt);
err:=(Havail-Hneeded)/abs(Havail);

//In case the error is small in the first step (i=0), the new temperature
//Titer, given by AFQuir will be far to small if 'dir:=err' is used.
if i>0 then dir:=err
else dir:=0.95;
AFQcode:=AFQUIR(Flowout.Tt, err, 0, 0.001, dir, 40,
'Reactor equilibrium iteration in' {+#13+
(Owner as TGSPcomp).CompIDstr}, Titer);
// if Titer<200 then Titer:=220; //If T<647.29, liquid water can exist,
if Titer<In1.Tt then Titer:=In1.Tt; //which is not accounted for in the
if Titer>6000 then Titer:=5000; //CombEquilibrium procedure.
Flowout.Tt:=Titer;
inc(i);
until (AFQcode>1) and (ConvergedAllowed);
if AFQcode=3 then Exit; //Error in AFQUIR iteration
StoichNOEq:=Gascompnewv[gtNO];
Result:=true;
end;

```

ViscosityPerSpecie (GSPglobal.Pas)

```
//This function calculates the viscosity for a specie using equation C.40.
function ViscosityPerSpecie(const T: double; const GC:TVisCoefArray): double;
    //Pay attention: viscosity is in (micropoise); in function
    begin    //'ReynoldsNumberIndex' the correction to kg/(ms) is performed.
        Result:=exp(GC[1]*LN(T)+GC[2]/T+GC[3]*IntPower(T,-2)+GC[4]);
    end;
```

VolumeComposition (GSPglobal.Pas)

```
//This function calculates the volume composition from the mass composition.
function VolumeComposition(const MassComposition:TGasComposition):TGasComposition;
var
    i          :TGas;
    invmolweight:double;
    GC         : TGasComposition;
begin
    invmolweight:=0;
    for i:=Low(i) to High(i) do if MassComposition[i]>NearlyZero then
        begin
            GC[i]:=MassComposition[i]/GasProperties[i].MoleMass;
            //Exclude water volume (assume contribution to volume not significant):
            if i<>gtH2O1 then invmolweight:=invmolweight+GC[i];
        end
    else GC[i]:=0;
    for i:=Low(i) to High(i) do Result[i]:=GC[i]/invmolweight;
    //Total of all GC components = 1+GC[gtH2O1].
    //GC[gtH2O1] is non-zero: this should be ignored; it is done to easily convert
    //the volume composition to the mass composition.
end;
```