

Optimisation of Combustor Wall Heat Transfer and Pollutant Emissions for Preliminary Design Using Evolutionary Techniques

J.M. Rogero P. A. Rubini*
Department of Automotive, Combustion and Energy Engineering,
School of Engineering
Cranfield University, MK43 0AL, Bedfordshire, UK

Abstract

This paper presents the conception and use of a design optimisation toolbox based on evolutionary techniques, for the preliminary design of gas turbine combustors. The toolbox has been designed to be interfaced with existing analysis packages and to perform optimisation in parallel over a network. The combustor design optimisation capabilities of the toolbox are demonstrated by automating the achievement of twenty-two performance targets for a combustor design whilst performing minimisation of wall cooling flow and NO_x emissions.

1 Introduction

Engineering design offices face a transitional phase in their working methods, comparable to the arrival of automation in the manufacturing industry.

The increasing complexity and performance requirements of engineering products means designers are faced with more decisions in the design phase than ever before, and the number and complexity of these decisions is growing rapidly.

In order to offload the designer, many of these decisions can be automated by integrating the simulation capabilities and the performance analysis with an optimisation tool. The automation of a number of design decisions using design optimisation codes has the potential to reduce the design lead times and costs while increasing the performance of the designs¹.

1.1 Status of design optimisation in the industry

Manual design and optimisation typically consists of taking an existing design and adapting it to a new problem by appropriate scaling and slight modifications. Optimisation is then achieved using mostly past experience (rule of thumb) and trial and error, optionally employing user driven computational analysis programs recursively until a suitable solution is found. This is particularly the case for gas turbine combustor preliminary design where new designs are mostly based upon previous designs, the experience of the engineer and trial and error.

@ British Crown Copyright 2001/DERA Published by the American Institute of Aeronautics and Astronautics, Inc.
with permission of the Controller of Her Majesty's Stationary Office

*Corresponding author

Complicated real-life designs are usually comprised of a large number of strongly correlated design variables with conflicting performance targets leading to challenging optimisation problems. The application of traditional techniques to these problems is difficult, resulting either in a lengthy and costly design phase or poor optimisation if time and resources are not available. The use of optimisation algorithms has the potential to tackle these design problems by reducing the amount of manual work towards obtaining a suitable optimised design. A lot of research has been devoted towards the generation of optimisation techniques usable for engineering design²⁻⁴.

The Microprocessor industry has already started the transition towards automation of the design process⁵. However many others industries continue to be behind in this domain. Recently by a survey of British industries⁶ highlighted the fact that optimisation algorithms are not yet widely used in design offices.

Several inhibiting factors are responsible for this limited usage:

- * Lack of integration of existing optimisation tools.
- * Limited optimisation skill among design engineers.
- * The computational cost of simulation.
- * The designer wants control over the optimisation.
- * The complexity of real life optimisation problems.

These inhibitors need to be overcome by the optimisation package for it to be accepted in the industry.

1.2 Gas turbine combustor preliminary design

1.2.1 Background

The preliminary design of a gas turbine combustor is concerned with shaping the basic features of the combustor to suit the capacity and performances requirements of the gas turbine. The required steps for the preliminary design of a combustor can be simplified as shown in Figure 1.

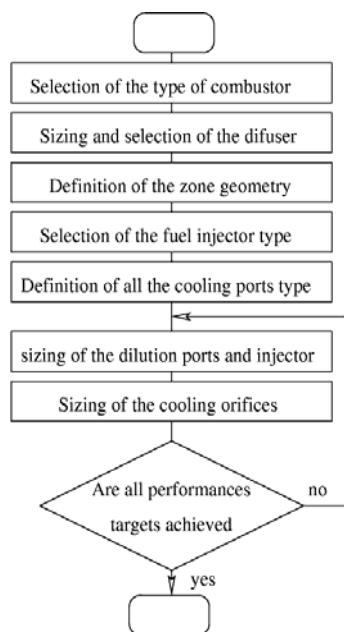


Figure 1: Design Process Flowchart

The performances targets for combustor design usually consist of: wall temperature, blow-off limits, combustion efficiency, emissions levels, relight performance, pressure drop, and outlet temperature traverse⁷. The design process consists of selecting the type of combustor, sizing it and finally tuning the design parameters by recursively performing simulations and modifications until all the performance targets are satisfactory. Such simulations are mostly based on semi-empirical mathematical models and correlations allowing rapid simulation of the combustor model.

The preliminary design process is typically followed by more detailed CFD analysis and rig testing to validate and refine the final design before manufacture of an engine demonstrator.

1.2.2 The need for improvement

Preliminary design is a demanding and time consuming processes relying a lot on past experience, empirical design rules, and trial and error. This process is becoming harder as on one side regulation⁸ and customers require increased performance, on the other side companies require a reduction in design costs and cycle time to stay competitive.

The use of optimisation algorithms have the potential to reduce the amount of designer work towards obtaining a suitably optimised design⁹⁻¹¹.

The technique proposed is a new and unique approach; it represents the first step towards autonomous engineering design with user-specified characteristics and objectives. This novel concept is

based on genetic algorithms (GA) supported by a set of modular tools necessary for engineering design¹².

A graphically based engineering optimisation toolbox has been developed, to ease the preliminary design of gas turbine combustors. It consists of a modular framework grouping a set of tools and operators required for engineering design.

2 Gas Turbine Combustor Preliminary Design Optimisation Toolbox

2.1 Problem approach

The aim of the toolbox is to provide a set of methods and tools suitable for engineering design. Which will allow the designer to optimise numerous parameters of a complex problem using the designer's traditional simulation software to evaluate the solutions. To be useful the tool needs to be as efficient as possible and achieve results in a reasonable amount of time. It also needs to be flexible to permit improvements of existing methods and the addition of new tools. Finally to be used in real life it needs to be user friendly and allow the designer to interact with the optimisation process without requiring detailed training in the optimisation domain.

In order to achieve these objectives it is desirable to use a modular object oriented architecture that will permit tools to be added thereby enhancing versatility and performance. As well, to allow the engineer to use his traditional analysis software, an interfacing tool able to communicate with a wide range of third-party software is advantageous. In order to be applied to engineering problems the optimisation tools need to be robust and efficient on a wide range of problems.

Genetic algorithms are used in this work for the main optimisation tool because of their robustness. But these need to be adapted to their use in engineering design to maximise the performance of the optimiser. As well, to reduce the computational overhead of simulation it was found necessary to design a tool distributing the evaluation of the analysis code over a network of heterogeneous computers.

The last part of the toolbox concerns the program/user interaction. The addition of a user friendly graphical interface where the designer can follow the evolution of all the problem parameters during the optimisation allows the user to get a 'feeling' of the optimisation process without requiring a detailed knowledge of optimisation.

2.2 Modular design

A significant degree of modularity was required in order to meet the desired aims of versatility and expandability of the toolbox. This was achieved by the use of Java as the main programming language,

being object oriented it readily allows development of this type of modular architecture. As well its platform independence is advantageous when working on a heterogeneous set of computers. Another useful feature of Java is the advanced support for networking and graphics. However this language suffers from being relatively slow compared to C/C++ and as well it is not memory efficient.

The toolbox is to be used for engineering design where simulation is performed by external analysis. Since the simulation process takes a very high proportion of the computational time the relative slowness of Java will not be a handicap, in addition the capacity of modern computer systems do not place a significant constraint on memory usage.

The object-oriented design allows modules to be loaded at runtime depending on the toolbox requirements. These modular tools are organised to support the interchangeable optimisation modules, by providing functionality such as for example, interfacing with other software, distributing the evaluation over a network. The optimisation modules are themselves composed of interchangeable sub-modules. These implement the basic functions of the optimiser. Figure 2 shows an example of the toolbox modular organisation.

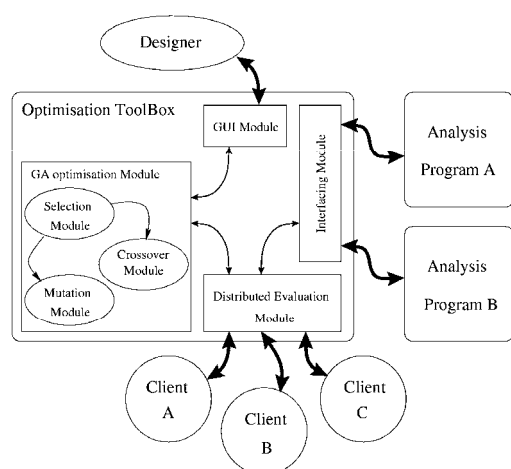


Figure 2: Modular Architecture

2.3 Adaptation to engineering design

In order to overcome the inhibitors mentioned in section 1.1, the optimisation toolbox has been carefully adapted to the problem of real life engineering design.

For the optimisation algorithm robustness has been given special importance by using a method based on genetic algorithms¹³ modified to be adapted to engineering design. See section 2.3.1.

One of the key factors defining the usability of the toolbox is its integration capability with the software

used for simulation. To allow easy integration of simulation capability within the toolbox an interfacing module has been developed and is described in section 2.3.2.

The performance of the evaluation is another key factor, simulation of engineering designs is usually costly in terms of CPU time. In order to overcome this problem a distributed evaluation module has been designed. See section 2.3.3.

The final factor is the need for the engineer to control the optimisation without in depth knowledge of optimisation techniques whilst tackling the complexity and the large number of parameters defining the quality of engineering design. This problem was solved by using a system able to deal with a high number of performance parameters, for which the user can define targets, validity range, and minimisation or maximisation of the parameter¹⁴.

2.3.1 The optimisation algorithm

This subsection describes our implementation of the Genetic Algorithms (GA). These, first developed by Holland¹⁵, provide a robust global optimisation method while being able to handle high-dimensional problems. For these reasons GA poses a good potential for design¹¹ and have been successfully applied to many engineering problems¹⁶⁻¹⁸.

The GA optimiser module is based on SGAJ from Hartely¹⁹, which is a Java implementation of the "simple GA" (SGA) from Goldberg's¹³. It has been extensively modified to adapt it to engineering design problems and maximise performance.

Traditional implementations of SGA use binary encoding to describe the optimisation parameters. However the work of Janikow & Michalewicz²⁰ suggests that coding the parameters using real numbers can improve the optimisation performances, especially for high dimensional problems. The chromosome modules have been modified to support real numbered parameter encoding.

The performances of real coded GA can be further improved by using the linear crossover operator introduced by Davis²¹. A module implementing this linear crossover has been implemented.

The adaptation of the mutation operator to real numbers is performed by implementing the decayed creep mutation operator. This operator is based on the real number creep mutation from Davis²¹. In our implementation, the operator adds a decay rate. This decay reduces the mutation range as the generations increase. It results in improved exploration capabilities at the start of the optimisation and fine local searches towards the end.

For engineering optimisation the evaluation of the fitness function requires calls to analysis codes, which are time consuming. In order to reduce the

numbers of call to the evaluation function Davis²¹ suggests to prevent the creation of duplicate genes, it even improves the performance of the GA. This duplicate prevention method has been implemented in the selection modules. In an effort to further reduce the number of evaluations Steady State replacement without duplicates has been implemented as well. This module has been extended to prevent the re-evaluation of genes by using a database of all the genes generated during the optimisation.

A random search phase was introduced at the beginning of the optimisation process to perform a broad exploration of the problem domain. This method was implemented in an attempt to improve the quality of the initial population. Moreover for design problems with constraints due to limitation of the simulation code, it allows to select only feasible design in the initial population after the random search.

2.3.2 Interfacing with simulation software

One of the main requirements of the Toolbox is the capacity to be interfaced with a wide variety of existing simulation software, in order to allow the application of the toolbox to a range of different problems. Another requirement is ease of use, requiring the user to program a specific interface to their analysis code is unacceptable.

One common feature of a majority of simulation codes, is their capacity to communicate through text files. It was decided to take advantage of this feature and perform communication with such codes through text files. The most robust way to locate the relevant data within an I/O text file was found to be the use of regular expression²² combined with the tokenisation of the file. This technique consists in searching data from a keyword within the file or from its position (line /column) or any combination of the two. Making it a very robust way to access data for read or write.

The interface module only requires the user to input the data location in the problem configuration file. For example data retrieval could be of the type: in the "node" section of the result file search for node "456" which is on the first column and read the data on the second next column.

2.3.3 Distributed computing

A very efficient way to reduce the computational overhead of recurrent runs of CPU intensive simulation software is to perform the execution of the applications within a Heterogeneous Distributed Computing²³ environment. The principle of this approach is to distribute the execution of individual processes over a network of computers which might not all be of the same architecture.

In the case of the distribution of multiple evaluations of the fitness function through the execution of a sequential analysis code, communication only takes place between the GA and the analysis software. This communication is performed through text files rather than inter-process communication alleviating the need to use, for example, MPI or PVM libraries.

2.3.4 Control of the performance parameters

One of the main difficulties to be solved was the large number of performance parameters that typically define the quality of engineering designs. This is especially true for combustor preliminary design, which usually requires between 20 and 50 performance parameters. In addition it is also necessary to have a straightforward way for the designer to interact with those parameters.

This problem was solved by developing a unique method where the designer can define for each parameter a target to be attained, a range this parameter should stay within, and the capacity of selecting the parameter for maximisation or minimisation. The quality of the design is then calculated from achievement of the targets, the possible violation of ranges, and the optimisation of the selected parameters.

This approach gives the designer total control over the optimisation without having to know too much about optimisation techniques and without having to devise a fitness function himself.

2.4 Modelling

The gas turbine combustor simulations were obtained using the flow simulation code Flownet. This code has been developed by Stuttford and Rubini²⁴. It is capable of modelling geometrical features of a gas turbine combustor to return information on temperature, pressure, mass flow-splits, fuel air ratios, and flow velocities, in different regions of the combustor.

Flownet is a network solver based on a semi-empirical method. The combustor is divided into a series of one-dimensional sub-flows containing independent semi-empirical governing equations. These equations are selected depending on the geometrical feature modelled. An overall governing equation links each sub flow to obtain a complete solution. This enables solutions to be obtained very rapidly in comparison to CFD. A detailed explanation of the network algorithm and Flownet can be found in ref²⁵.

In order to give some information to the optimiser about the NO_x emission of the combustor, a simple NO production model based on the Zeldovich²⁶ mechanism was incorporated. The model calculates the NO production rate from the equilibrium concentration of O₂ and N₂ using mean zone

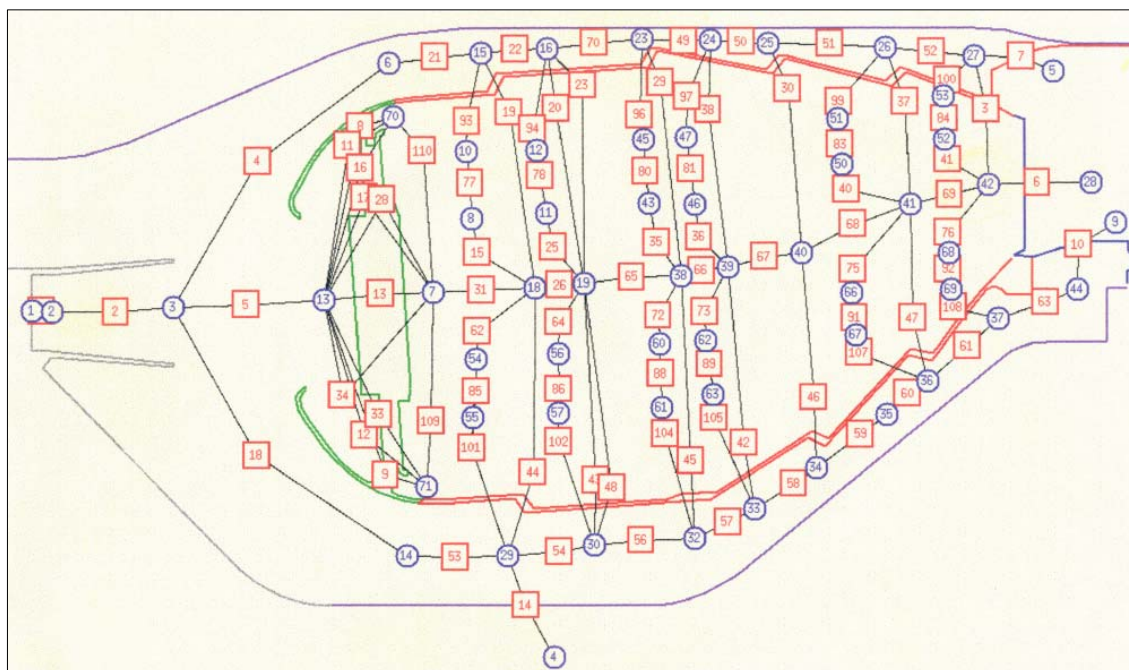


Figure 3: Network Model of the Combustor

temperatures and zone residence times. Total NO is obtained by summing all elements along the flow path. It is important to note that this is only a basic model of a complexity somewhere between that of an empirical correlation and a stirred reactor²⁷. It is not quantitatively accurate but should be suitable to demonstrate the trend of the NO_x production depending upon modification of the combustor parameters. This model is described in detail in ref¹⁴.

3 Application of the optimisation toolbox

In order to demonstrate the design automation and optimisation capabilities, the optimisation toolbox has been applied to the preliminary design of a conventional 1970's combustor design.

3.1 Preliminary design of a single annular combustor

The design test case is of an existing single annular aero-engine combustor. A model of this combustor was made for the Flownet simulation code, see Figure 3. Combustor flame tube cooling is achieved through effusion ports and z-rings and the combustion is controlled by 2 sets of dilution ports. The geometric characteristics were encoded into twenty-five real-numbered chromosomes. The alternative combustor networks generated by the optimiser were evaluated using the network simulation code.

3.1.1 The optimisation problem

The quality of a combustor design is defined by a number of performance parameters. Ideally these

performance parameters should be: the overall pressure drop, the pollutant emission, the relight altitude, and the exit temperature profile with a constraint on the maximum wall temperature. However the preliminary design tools are not able to predict the exit temperature traverse, the soot, the CO, and the UHC emissions. So additional constraints were added to indirectly control these performance parameters.

* In order to control mixing which affects the pollutants emissions and the exit temperature traverse some constraints are put on the overall, and flame tube wall pressure loss, as well as on the port cross follow Mach number.

* The relight altitude is controlled by a relight loading parameter.

In total those sums up to 22 performance targets to control. With these defined there are two ways to use the optimisation tool.

* One is to use it as a design automation tool and to achieve predefined values for the performance parameters as demonstrated in section 3.12.

* The other is to use it as an optimisation tool and set it to minimise or maximise parameters while constraining other parameters to predefined targets demonstrated in section 3.1.3 and 3.1.4.

To perform these tasks the toolbox is set-up to modify all the combustor cooling and dilution holes, which define the flow within the combustor. A number of 25 genes encodes and controls the size of the ports, the effusion patches, the Z-ring, the fuel injector, and the base plate cooling orifices. At the

moment the geometry of the combustor is not modified. The performance parameters are calculated from 140 result values extracted from the Flownet output file.

3.1.2 Designing for targets

This first task consisted of mimicking the manual design by taking exactly the same performances targets as the manually designed combustor. For all of the 22 performance parameters an allowable range was set to ensure the validity of the design. This range was of $\pm 5\%$ for the critical parameters and $\pm 10\%$ for the less critical parameters.

The optimisation was performed in 4h15min using a single workstation for 10000 evaluation; all the targets were within the allowable range after 2700 evaluations.

The results of the optimisation are shown in Table 1. From this table it is possible to see that all the critical parameters were achieved with a high precision and that all the targets parameters were achieved with less than 3.5% error.

3.1.3 Designing for minimum cooling flow

This second task consisted of using the optimiser to achieve the same targets as in section 3.1.2 but with the added aim to reduce the amount of flow used to cool the flametube walls.

The optimisation was performed in 4h 30min using a single workstation for 10000 evaluation, all the targets were within the allowable range after 2500 evaluations.

After 10000 evaluations the optimiser managed to reduce the amount of cooling flow by 23.2%

The results of the optimisation are shown in Table 1. From this table it is possible to see that all the parameter were still achieved within their allowable range of error. Figure 4 shows the evolution of the cooling flow with the advancement of the optimisation, it can be seen that no optimisation is performed until all the targets are within the allowable range. As well Figure 5 shows the flow through the cooling orifice for the manual design and for the optimised design.

3.1.4 Designing for minimum NOx

The final task was to demonstrate the capability of the optimiser to optimise the design of the combustor for NOx emissions.

The optimisation was performed in 36 minutes using 10 networked workstations for 10000 evaluation; all the targets were within the allowable range after 2000 evaluations. The optimiser managed to reduce the value of the predicted emission by 18.6%

The results of the optimisation are shown in Table 1. As before it can be seen that all the parameter were achieved within their allowable range of error.

Figure 6 shows the evolution of NOx prediction with the advancement of the optimisation, and Figure 7 shows the relight-loading factor for the combustor. From those two graphs it is interesting to note that the reduction of NOx is stopped when the relight-loading factor hits the limit of its range. This shows that the relight loading which is dependent of the combustor mass flow is a limiting factor.

	Design For Targets	Minimise cooling	Minimise NOx
Pressure drop comb	Range $\pm 5\%$ 0.27%	Range $\pm 5\%$ -0.49%	Range $\pm 5\%$ -0.39%
Pressure drop wall 1	Range $\pm 5\%$ 0.43%	Range $\pm 5\%$ -0.80%	Range $\pm 5\%$ -0.63%
Pressure drop wall 2	Range $\pm 5\%$ 0.41%	Range $\pm 5\%$ -0.75%	Range $\pm 5\%$ -0.60%
AFR injector	Range $\pm 5\%$ -2.82%	Range $\pm 5\%$ 3.69%	Range NA 12.80%
AFR 1	Range $\pm 5\%$ 0.56%	Range $\pm 5\%$ 0.67%	Range NA 4.82%
AFR 2	Range $\pm 5\%$ 0.90%	Range $\pm 5\%$ 2.00%	Range NA 4.16%
Flametube cooling	Range $\pm 5\%$ -1.90%	Range NA -23.28%	Range $\pm 10\%$ -1.67%
Base Plate Cooling	Range $\pm 5\%$ 1.42%	Range $\pm 5\%$ -4.97	Range $\pm 5\%$ 4.69%
Max Temp Combustor	Range +0% -0.02%	Range +0% -0.70%	Range +0% -0.10%
Max Temp Zone 1	Range +0% -3.00%	Range +0% -0.89%	Range +0% -1.95%
Max Temp Zone 2	Range +0% -0.02%	Range +0% -0.70%	Range +0% -0.10%
Avg Temp Combustor	Range +5% 0.15%	Range +5% 1.22%	Range +5% 1.08%
Avg Temp Zone 1	Range +5% -0.54%	Range +5% -0.73%	Range +5% -1.21%
Avg Temp Zone 2	Range +5% 0.75%	Range +5% 2.34%	Range +5% 2.39%
Recirculating flow	Range $\pm 5\%$ 0.31%	Range NA 1.84%	Range NA 4.97%
SI Loading	Range $\pm 5\%$ 0.03%	Range $\pm 5\%$ -0.05%	Range $\pm 5\%$ -0.04%
Relight loading factor	Range $\pm 5\%$ 0.29%	Range $\pm 5\%$ 1.82%	Range $\pm 5\%$ 4.99%
NOx	Range +0% -1.21%	Range +0% -5.97%	Range +0% -18.59%
Mach ratio Outer Ports 1	Range $\pm 10\%$ -1.89%	Range $\pm 10\%$ -5.36%	Range $\pm 10\%$ -1.06%
Mach ratio Inner Ports 1	Range $\pm 10\%$ 0.73%	Range $\pm 10\%$ -0.17%	Range $\pm 10\%$ 8.10%
Mach ratio Outer Ports 2	Range $\pm 10\%$ -0.93%	Range $\pm 10\%$ 1.17%	Range $\pm 10\%$ 3.91%
Mach ratio Inner Ports 2	Range $\pm 10\%$ 3.33%	Range $\pm 10\%$ -2.02%	Range $\pm 10\%$ 7.21%

Table 1: allowable range and target achievement of the three design optimisation cases

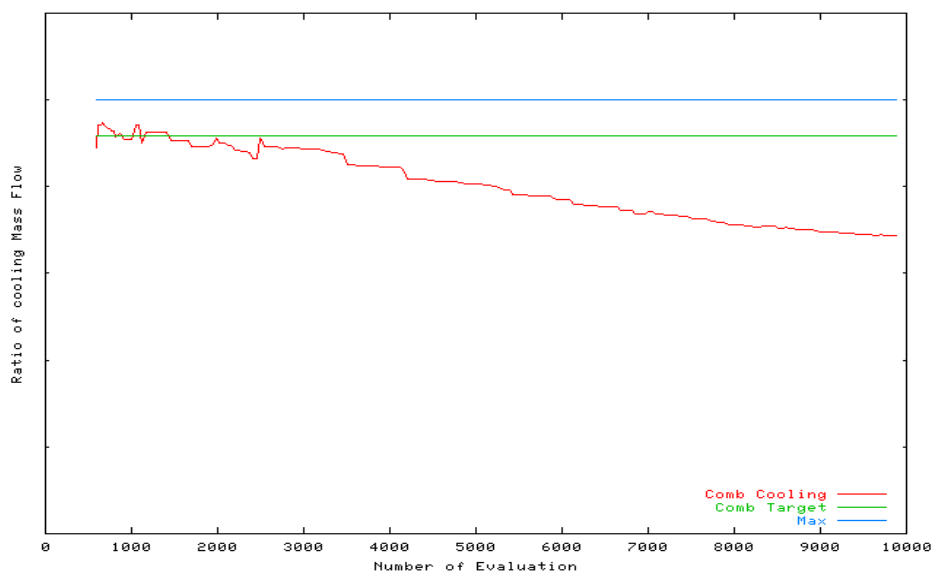


Figure 4: Evolution of the cooling flow

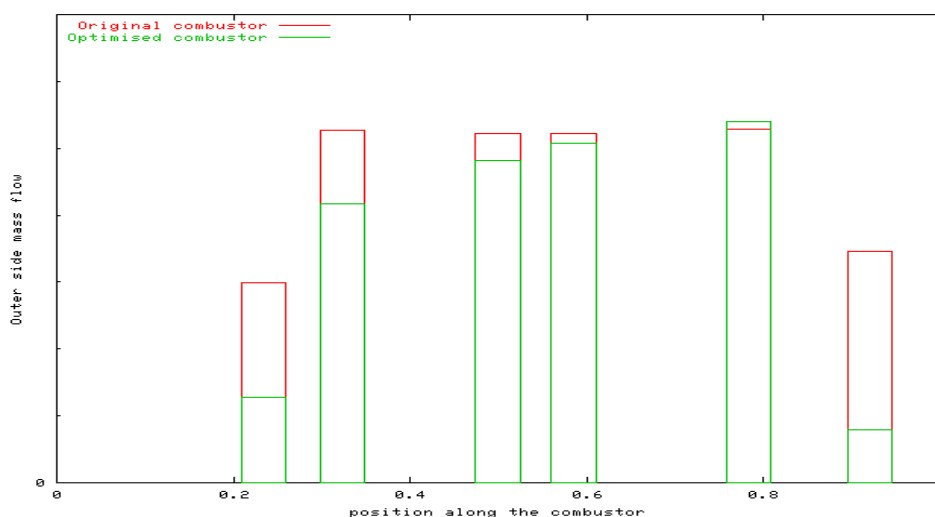


Figure 5: Amount of cooling flow along outer wall of combustor

Figure 8 shows the AFR along the flame-path for the manual design and the optimised design. From this figure it can clearly be seen that the optimiser went for a leaner primary zone.

And finally Figures 9-11 shows the evolution of the AFR of respectively, the injectors, and in the zone and 2 of the combustor. From those graphs it is interesting to note that the optimiser chooses to go lean to reduce the NO_x emissions.

4 Conclusion

The results obtained for the automation of design and the optimisation of design performance using the

optimisation toolbox demonstrated the capacity of this type of tool to be used for real life problems involving a large number of conflicting performance parameters as well as a large number of problem variables.

The optimiser showed its capacity to tune the performance of combustor during the preliminary design phase. As well the time taken to achieve these results, in the order of a few hours is well below than for manual design, which is in the order of a few weeks or months.

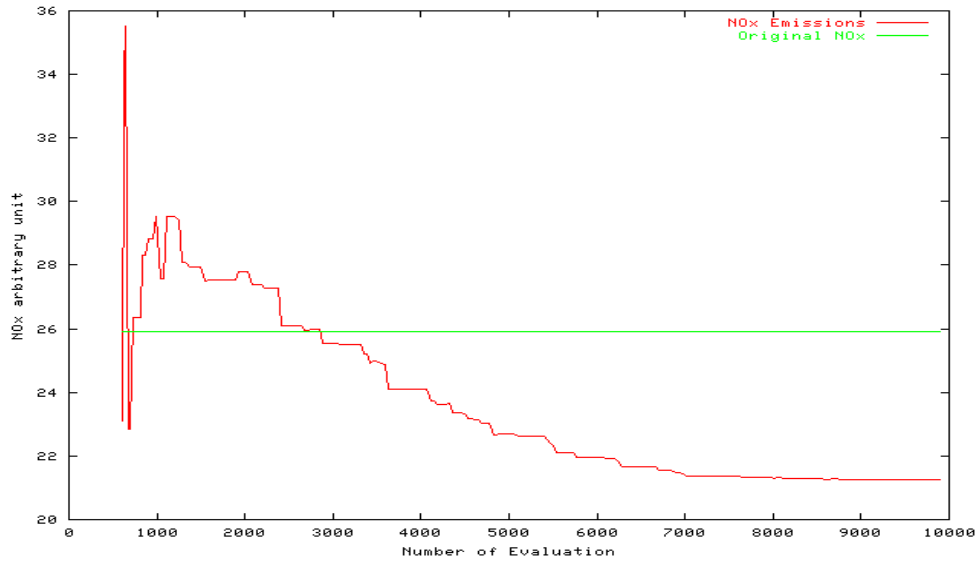


Figure 6: Predicted NOx Emissions (Arbitrary Units)

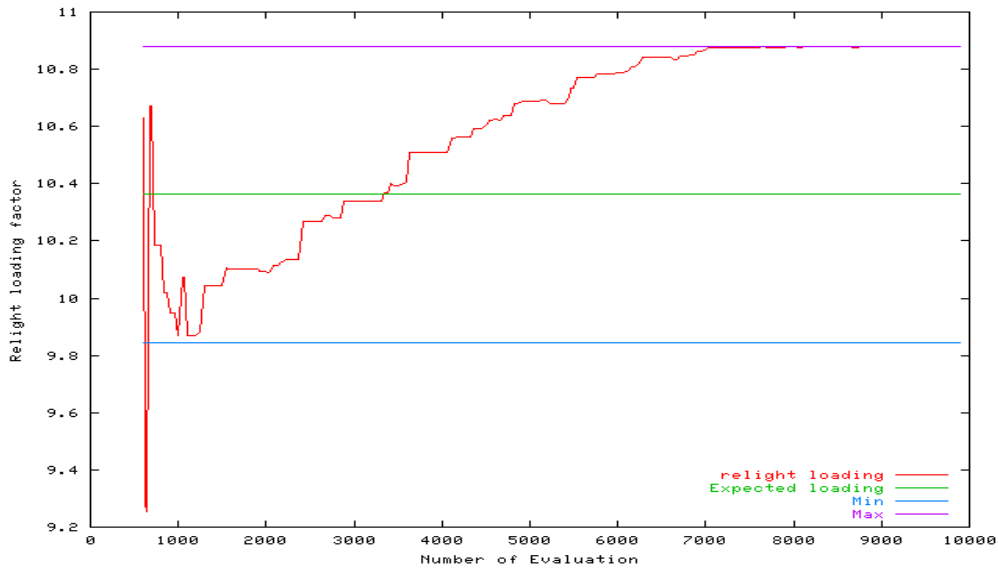


Figure 7: Relight Loading Factor

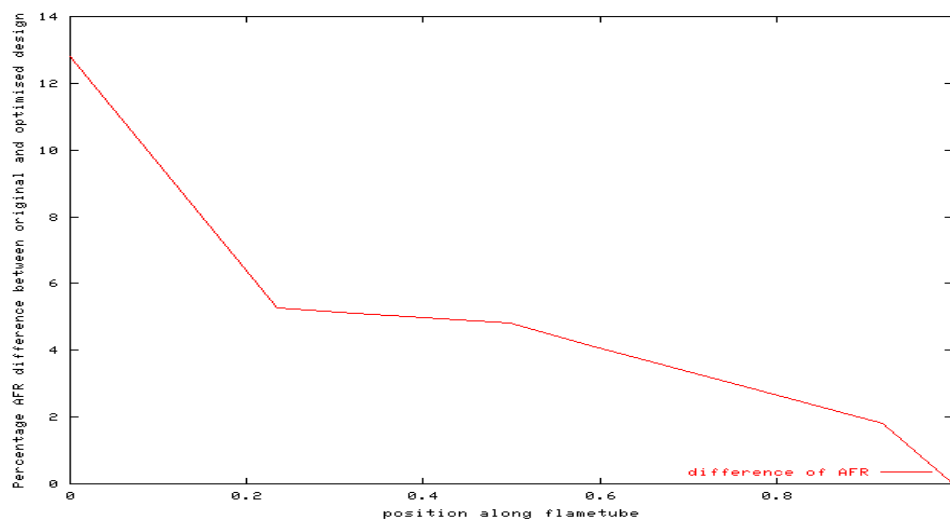


Figure 8: Difference Of AFR along the combustion stream original/optimised design

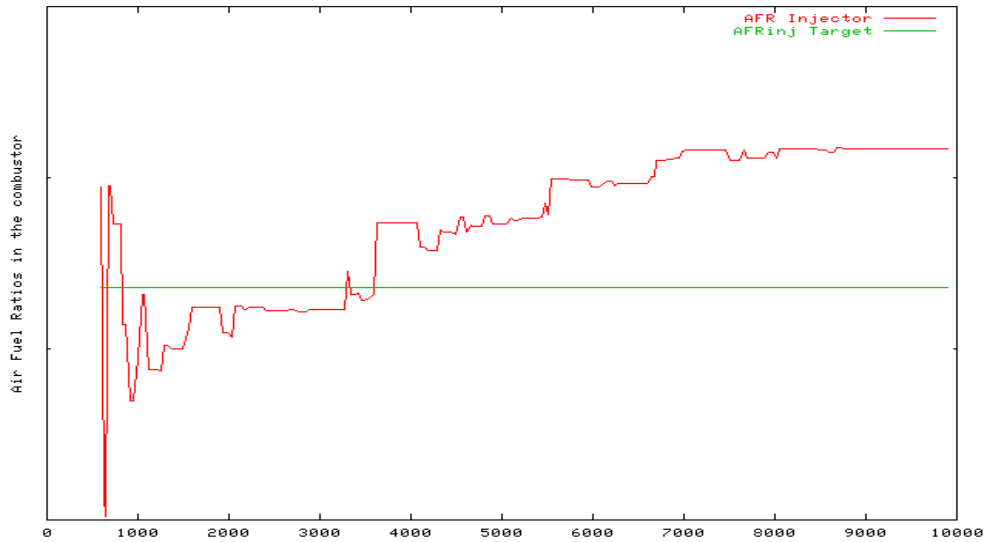


Figure 9: AFR Injector

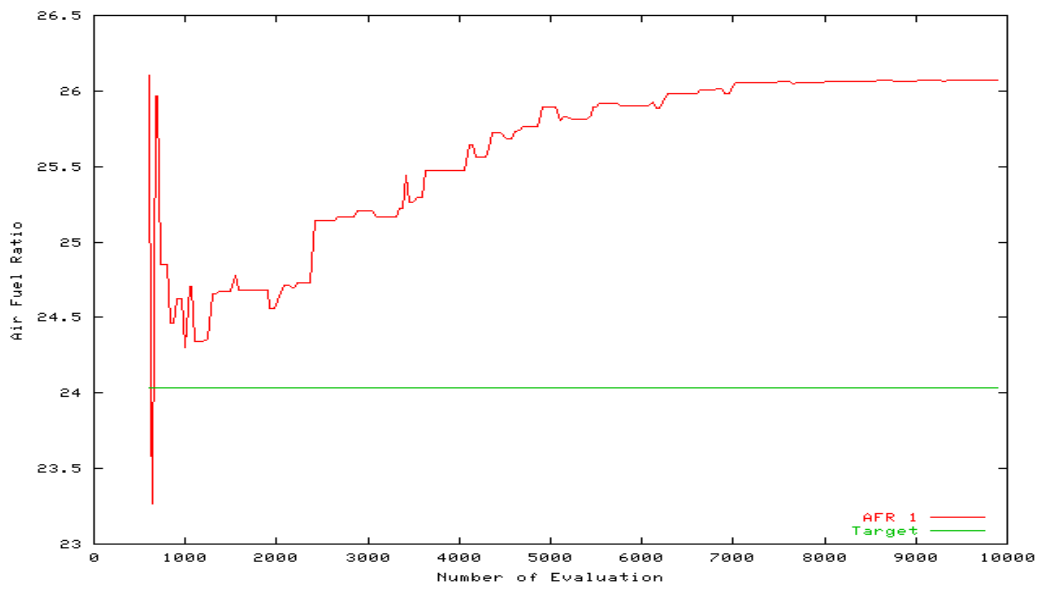


Figure 10: AFR 1 (exit of Primary zone)

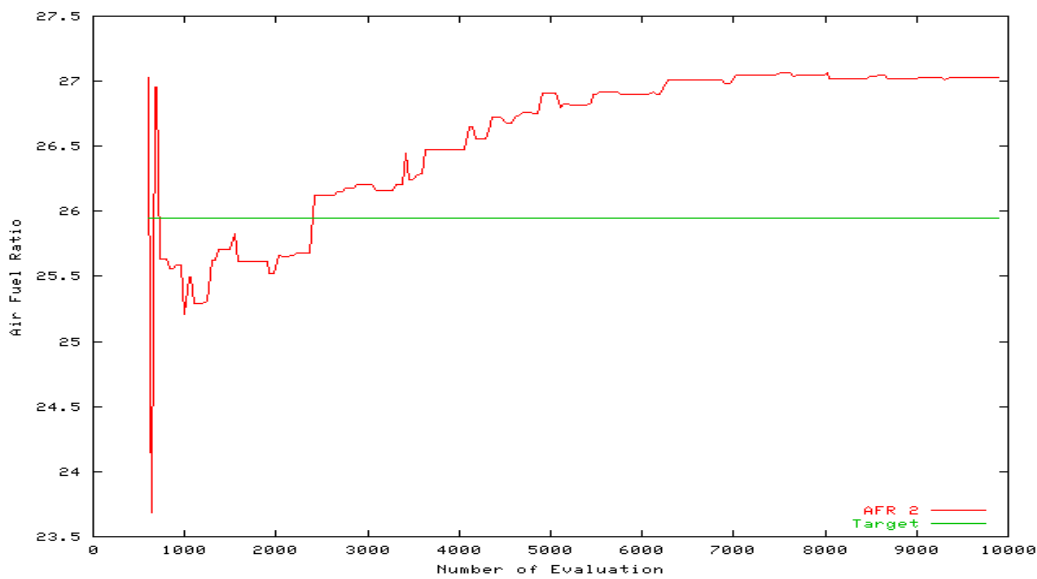


Figure 11: AFR zone 2 exit

Acknowledgements

The authors wish to acknowledge the grateful support of the Defence Evaluation and Research Agency (DERA) and Rolls Royce plc.

References

- [1] J. M. Rogero et al, Applications of evolutionary algorithms for solving real life design optimisation problems, Sixth conference on parallel problem solving from nature, workshop proceedings, (pp85-87), 2000.
- [2] P.M. Pardalos et al, Recent developments and trends in global optimisation, Journal of computational and applied Mathematics, Vol. 124, (pp209-228), 2000.
- [3] J. Sobieszczanski-Sobieski and R. T. Haftka, Multidisciplinary aerospace design optimization: survey of recent developments, Structural Optimization 14, (pp. 1-23), Springer-Verlag, 1997.
- [4] G. N. Bullock et al, Developments in the use of genetic algorithm in engineering design, Design studies Vol. 16, (pp507-524), 1995.
- [5] W. M. Johnson, Superscalar Microprocessor Design, Prentice-Hall, Englewood Cliffs, NJ, 1991.
- [6] Dr R. Roy et al, Design Optimisation: A Survey of British industries. Flexo Report 5, SIMS, Cranfield University, 2000.
- [7] A. H. Lefebvre, Gas turbine combustion second ed., Taylor & Francis, ISBN 1-56032-673-5, 1999.
- [8] M. Le Dilosquer, The aero engine response to the protection of the global atmosphere, IMechE seminar on Gas Turbine Pollutant emissions Technology Advances and Updates, 1999.
- [9] K. Deb and M. Goyal, Optimising engineering designs using a combined genetic search. Proceedings of the Seventh international conference on genetic algorithms, (pp. 521-528), Morgan Kaufman, 1997.
- [10] B. Esping, Design Optimisation as an engineering tool. Structural Optimisation 10, (pp. 137-152), Springer-Verlag, 1995.
- [11] I. C. Parmee, Exploring the design potential of evolutionary / adaptive search and other computational intelligence technologies. Adaptive Computing in Design and Manufacturing. (pp. 27-44), Springer-Verlag, April 1998.
- [12] J. M. Rogero and P. A. Rubini, A platform independent engineering optimisation tool based on genetic algorithms and distributed computing applied to gas turbine combustor preliminary design, computational engineering using metaphors from nature, B. Topping, Civil-Comp Press, ISBN 0-948749-66-0, (pp143-149), 2000.
- [13] D. E. Goldberg, Genetic algorithms, in search of optimisation & machine learning, Addison-Wesley Publishing, 1989.
- [14] J. M. Rogero, Preliminary design of gas turbine combustor using genetic algorithms, PhD thesis, Cranfield University UK, to be submitted 2001.
- [15] J. H. Holland, Adaptation in natural and artificial systems, Ann Arbor, University of Michigan Press, 1975.
- [16] R. A. E. Makinen et al, Multidisciplinary shape optimisation in aerodynamics and electromagnetic using genetic algorithms, International Journal For Numerical Methods In Fluids 30, (pp149-159), 1999.
- [17] A. M. Trulove and K. W. Whitaker, Rocket Stage Optimization Using a simple genetic algorithm, AIAA-93-1778, 29 Joint Propulsion Conference and Exhibit, Monterey, CA, June 1993
- [18] R. V. Tappeta et al, A multidisciplinary design optimisation approach for high temperature aircraft engine components, Structural optimisation, Vol. 18, (pp 134-145), 1999.
- [19] S. J. Hartley. Concurrent programming the Java language. Oxford University press, 1998.
- [20] GC. Janikow and Z. Michalewicz, An experimental comparison of binary and floating point representations in genetic algorithms, Proceedings of the fourth international conference on genetic algorithms, (pp. 31-36), Morgan Kaufman, 1991.
- [21] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.
- [22] D. E. Knuth, The art of computer programming, fundamental Algorithms, 2nd ed., vol. 1, Addison-Wesley, 1973.
- [23] V. S. Sunderam and G. A. Geist, Heterogeneous parallel and distributed computing. Parallel Computing 25, (pp. 1699-1721), Elsevier, 1999.
- [24] P. J. Stuttaford and P. A. Rubini, Preliminary gas turbine combustor design using a network approach, ASME paper 96-GT-135, 1996.
- [25] P. J. Stuttaford, Preliminary gas turbine combustor design using a network approach, PhD Thesis School Of Mechanical Engineering Cranfield University, UK, 1997.
- [26] J. Zeldovich, Acta Physiochimica, Vol.21,(pp 577), 1946
- [27] J. B. Moss. Predictive methods for gas turbine combustor emissions, IMechE seminar on Gas Turbine Pollutant emissions, 1999.