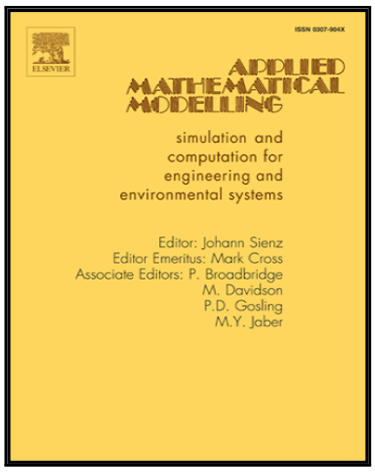


# A novel implementation of computational aerodynamic shape optimisation using Modified Cuckoo Search

D.S. Naumann, B. Evans, S. Walton, O. Hassan

PII: S0307-904X(15)00737-4
DOI: 10.1016/j.apm.2015.11.023
Reference: APM 10887

To appear in: *Applied Mathematical Modelling*

Received date: 24 March 2015
Revised date: 15 October 2015
Accepted date: 10 November 2015

Please cite this article as: D.S. Naumann, B. Evans, S. Walton, O. Hassan, A novel implementation of computational aerodynamic shape optimisation using Modified Cuckoo Search, *Applied Mathematical Modelling* (2015), doi: 10.1016/j.apm.2015.11.023

**Highlights**

- a novel method of parameterising a computational mesh using 'control nodes'
- 'control nodes' coupled to the volume mesh via a Delaunay Graph Mapping technique
- Modified Cuckoo Search - an evolutionary, global optimisation algorithm - is employed
- applied to a range of test cases such as intake duct optimisation
- demonstrated to be effective in terms of its generalised applicability and intuitiveness

# A novel implementation of computational aerodynamic shape optimisation using Modified Cuckoo Search☆

D. S. Naumann[a,*], B. Evans[a], S. Walton[a], O. Hassan[a]

[a]*Civil & Computational Engineering Centre, School of Engineering, Swansea University, Swansea SA2 8PP, Wales, UK*

**Abstract**

This paper outlines a new computational aerodynamic design optimisation algorithm using a novel method of parameterising a computational mesh using 'control nodes'. The shape boundary movement as well as the mesh movement is coupled to the movement of user–defined control nodes via a Delaunay Graph Mapping technique. A Modified Cuckoo Search algorithm is employed for optimisation within the prescribed design space defined by the allowed range of control node displacement. A finite volume compressible Navier–Stokes solver is used for aerodynamic modelling to predict aerodynamic design 'fitness'. The resulting coupled algorithm is applied to a range of test cases in two dimensions including aerofoil lift–drag ratio optimisation intake duct optimisation under subsonic, transonic and supersonic flow conditions. The discrete (mesh–based) optimisation approach presented is demonstrated to be effective in terms of its generalised applicability and intuitiveness.

*Keywords:* mesh movement, Cuckoo search, computational fluid dynamics, aerodynamic shape optimization, shape parameterisation

## 1. Introduction

During the last 30 years, the aerodynamic design problems faced by the aerospace industry have been revolutionised by computational fluid dynamics (CFD). Particularly unstructured mesh methods [1, 2, 3] these days allow the mesh generation on complex three–dimensional geometries within a few hours, that initially required several months using multiblock techniques for quasi–structured meshes [4, 5]. Simultaneously, the development of Computer Aided Design (CAD) has had a strong impact on the design cycle of aerodynamic problems [6]. In light of this, CFD and CAD have become integral parts of a typical aerodynamic design cycle apparent in current aerodynamic design projects. The flow chart in figure 1 [7] indicates the emphasis now placed on CFD and CAD within the inner and outer design loops.

Despite these advancements, significant challenges remain for the computational modelling community in order to efficiently transfer geometry between CAD and CFD systems and improve the computationally expensive mesh re-generation process and CFD evaluation during optimisation [7, 8]. Main challenges include a lack of standardised shape parameterisation approaches and the alignment of CAD geometry definition with the CFD

☆The intellectual property rights in the research data are asserted by the authors and their research partners.

*Corresponding Author: Tel.: +447805243278

*Email addresses:* 717761@swansea.ac.uk (D. S. Naumann), b.j.evans@swansea.ac.uk (B. Evans), s.p.walton@swansea.ac.uk (S. Walton), o.hassan@swansea.ac.uk (O. Hassan)

[1]Abbreviations: FDGD - Fast Dynamic Grid Deformation, DG - Delaunay Grid, CS - Cuckoo Search, MCS - Modified Cuckoo Search

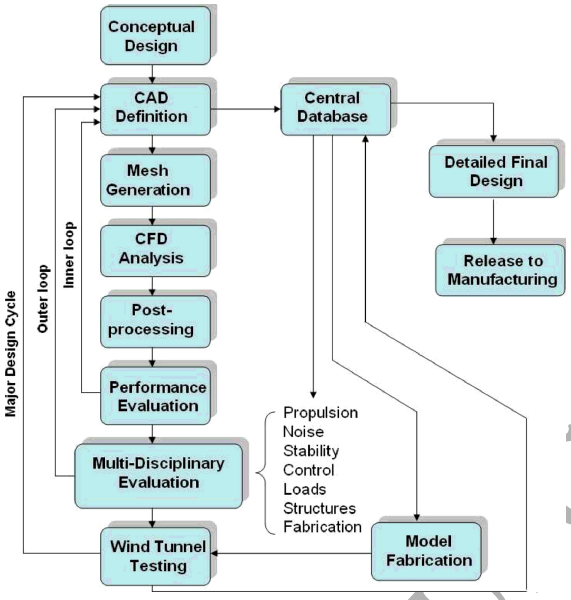


Figure 1: A typical multi–disciplinary aerospace design cycle

solver geometry definition as well as the lack of consensus regarding most suitable optimisation scheme given the application. Current research tries to overcome some of these problems by linking CAD systems, CFD tools and the mesh generation process. Examples include Isogeometric Analysis [9, 10, 11] and NURBS Enhanced Finite Elements Methods [12, 13], Reduced Order Models (e.g. POD) [14, 15] and surrogate models (e.g. Kriging) [16, 17]. These approaches have been shown to reduce the computational cost involved in the CFD evaluation. Analysis of the impact of these approaches was deemed to be beyond the scope of this study. However, limited progress has been made to link these approaches to computational aerodynamic shape optimisation algorithms in contexts of generalised applicability. The selected approach usually severely limits the explorable design space (i.e. range of potential shapes) for the optimiser. Considerable effort into research concerning coupling CFD modelling with aerodynamic shape optimisation has only been invested in significantly over the last 10 years [18]. Particularly the use of global optimisation algorithms in this field is only just emerging [19].

This paper presents a novel implementation of computational aerodynamic shape optimisation in which the parameterisation of the geometry and coupling with an optimisation algorithm is unique. The approach makes use of the concept of 'control nodes' in the mesh as the method for both defining the geometry movements and as the design parameters for the optimisation process. The Fast Dynamic Grid Deformation (FDGD) approach [20] has been applied to move the mesh and results in a self-contained algorithm formulated to propagate the effect of the 'control node' displacement throughout the discrete shape boundary and computational mesh. There is no requirement to re–mesh at each stage in the optimisation. Since all knowledege of the geometry is 'stored' in the discrete boundary, there is no requirement to convert the geometry definition stored in the mesh into any other format during the optimisation process. This reduces the problem of translation of CAD-based geometry definitions to CFD meshes.

Aerodynamic designers prefer to use tools that are both intuitive and have wide–ranging applicability. The

optimisation and design process requires an effective geometry parameterisation to allow sufficient exploration of a design space. Furthermore, a minimisation of the number of parameters defining the position in a design space is of benefit in order to also minimise computational cost. The approach described in this paper is a 'control node' mesh–based parameterisation. Well–known mesh–based optimisation test cases were performed by Jameson [21, 22] using control theory coupled with an adjoint approach[7, 22, 23] to solve for gradients. Other implementations of parameterisation schemes in the literature include CAD based [24], analytical, basis vector [25], free form deformation (FFD) [26, 27], domain element methods [28], and the control grid approach [29]. A thorough review of shape parameterisation techniques is provided by Samareh in [30]. In this paper it will be argued that the 'control node' approach presented here has advantages over these methods in terms of ease of implementation, user intuition and generalised applicability.

## 2. Methodology

### *2.1. Geometry Shape Parameterisation*

One of the common practical problems in industrial implementations of CFD–based aerodynamic design is the translation of geometries from CAD systems into computational meshes for simulation. This is often referred to as the bottleneck of the design process [7, 8] due to differing tolerance levels required for CAD systems compared with CFD [31, 32]. Solutions currently researched, for example Isogeometric Analysis and NURBS Enhanced Finite Elements Methods emphasize the development of a new CFD solver based on the geometry definition of the CAD system [9, 10, 11, 12, 13].

Once the initial computational mesh has been created (which could have originated as a CAD geometry), the geometry is then parameterised by choosing 'control nodes' at critical positions defined by the user on the discrete shape boundary. The number and position of these control nodes is important in determining how the geometry will evolve. Figure 2 shows the definitions of the terms 'control nodes', 'boundary nodes' and 'domain nodes' that will be used throughout this paper.

One of the important features of this parameterisation technique is that as the number of control nodes, $n_{cn}$, approaches the number of boundary nodes, $n_p$, the parameterisation approaches an unlimited scope of the potential design space for a given boundary discretisation, $\boldsymbol{B}$ , i.e. if $\boldsymbol{B} \in \Re^d$ then as $n_{cn} \rightarrow n_p$, $\boldsymbol{B} \rightarrow \Re^d$. $d$ describes the degrees of freedom within the system. Thus, the dimensionality of the explorable design space can be adjusted through the number of control nodes.

A general definition of the total degrees of freedom within the system is given by

$$d = n_{cn} f_{cn} \tag{1}$$

where $f_{cn}$ is the number of degrees of freedom per control node. Generally, for 2–D cases, $f_{cn}$ would be 2, however, in some of the case studies considered here, the $f_{cn}$ value was reduced to 1 (by restricting potential control node movement) to further reduce the design space dimensionality and, therefore, computational cost.

The explorable design space should be considered as a range of displacements from an initial discrete shape boundary, where the boundary deformation is a function of control node movement as,

$$\boldsymbol{B_{new}} = \boldsymbol{B_{init}} + \Delta\boldsymbol{B}(\Delta\boldsymbol{C_1}, ..., \Delta\boldsymbol{C_{n_{cn}}}) \tag{2}$$

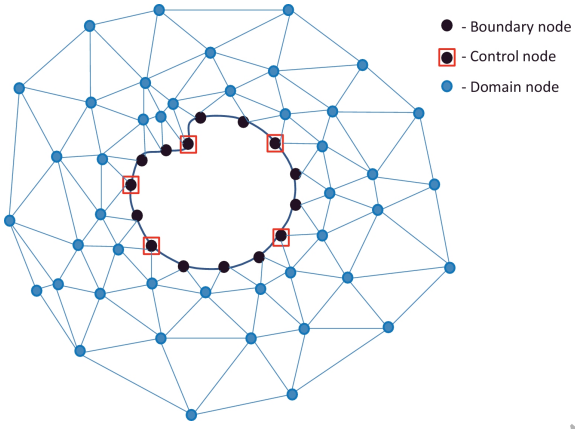


Figure 2: Definitions of control nodes, boundary nodes and domain nodes

where $\boldsymbol{B_{init}}$ is the initial boundary definition, $\boldsymbol{B_{new}}$ is the new boundary shape and $\boldsymbol{C_i}$ is the position vector of control node, $i$.

### *2.2. Mesh Movement*

#### *2.2.1. Background*

Significant challenges are involved with mesh movement techniques particularly in cases of large geometry deformation. Existing methods are commonly separated into two different categories: schemes based on mesh connectivity and point-by-point schemes [33]. Examples of the first category include the widely used spring analogy / solid body elasticity approaches [34, 35, 36, 37] which exploit the mesh connectivity but require the solution of large systems of equations resulting in high computational expense. Point-by-point approaches [33, 38, 39, 40] offer similar robustness at considerably lower computational cost by modifying the positions of the mesh nodes without making use of the mesh connectivity. Transfinite interpolation [38] and radial basis function interpolation [33, 39, 40] are common examples amongst others. The most promising point-by-point method investigated is based on Delaunay Graph (DG) Mapping called 'Fast dynamic grid deformation' (FDGD) [20]. This method was utilised as the basis for the mesh movement in the work presented in this paper.

#### *2.2.2. Control Node Approach for Mesh Movement*

The mesh movement is a three step process to propagate the initial movement of the control nodes first to the discrete boundary and then throughout the entire computational mesh. An overview of the scheme employed is provided in Figure 3. The optimisation algorithm dictates the displacement of the control nodes by analysing the fitness of each geometry (details are presented in Section 2.3). In the second step, the discrete shape boundary is deformed based on the 'control nodes displacement and utilizing a new formulation described in the subsequent chapter. Finally, the domain nodes are moved.

#### *2.2.3. Fast Dynamic Grid Deformation*

To move a mesh using FDGD five steps are employed: 1. generate a coarse 'background' Delaunay Graph (DG) using all boundary nodes. 2. locate the domain nodes in the coarse DG. 3. calculate the area coefficients

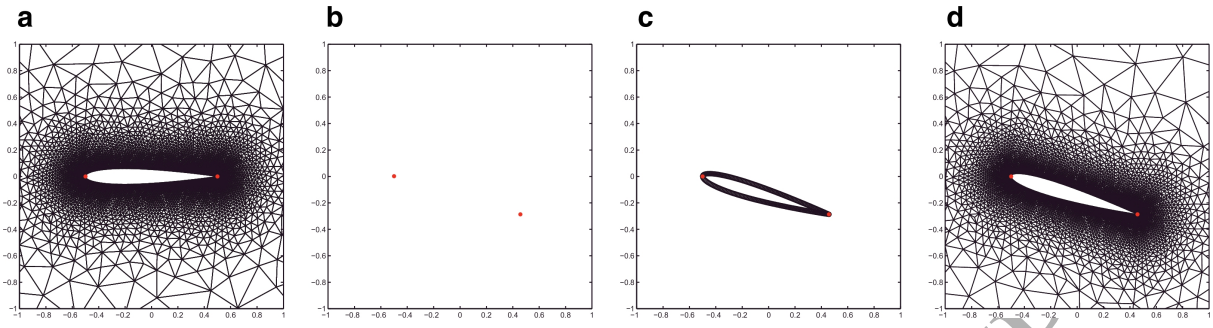


Figure 3: Mesh movement methodology: (a) Initial mesh provided as an input and starting point for the optimisation (b) the control nodes are relocated based on the MCS optimiser (c) the discrete shape boundary is moved (d) the domain nodes within the mesh are moved whilst retaining mesh connectivity Red points - control nodes

of each of the domain nodes. 4. move the DG according to the boundary change. 5. relocate the domain nodes by maintaining area coefficients [20].

In order to apply the technique for both moving the boundary nodes as well as the domain nodes, the methodology has been extended and a 'hyper-coarse' background mesh has been introduced. In effect, the methodology is duplicated to first move the boundary nodes based on the 'control node' displacement. The 'control nodes' and the corner nodes in the far field boundary are used to generate a DG termed the 'hyper-coarse background' DG. Next, the domain nodes are moved based on the boundary node movement by generating a DG using all boundary nodes. This is termed the 'coarse background' DG. Implementing this modification, the methodology becomes:

1. generate hyper-coarse background DG with 'control nodes' (Figure 4 (a)-(b))
2. locate the boundary nodes in the hyper-coarse DG
3. generate coarse background DG with boundary nodes (Figure 4(d)-(e))
4. locate the domain nodes in the coarse DG
5. calculate Area Coefficients of domain nodes in coarse DG
6. calculate Area Coefficients of boundary nodes in hyper-coarse DG
7. move the hyper-coarse DG according to the 'control nodes' displacement (Figure 4c)
8. relocate boundary nodes by maintaining Area Coefficients
9. move the coarse DG according to the boundary movement (Figure 4f)
10. relocate domain nodes by maintaining Area Coefficients

The mesh $\boldsymbol{M}$ is now solely influenced by the 'control node' displacements $\Delta C$ and can be described by the equation

$$\boldsymbol{M_{new}} = \boldsymbol{M_{init}} + \Delta \boldsymbol{M}(\Delta C_1, ..., \Delta C_{n_{cn}}) \tag{3}$$

where $\boldsymbol{M_{init}}$ is the initial mesh, $\boldsymbol{M_{new}}$ is the new mesh.

Apart from its known effectiveness, the method also benefits from very good shape preservation characteristics for deformations with equal 'control node' displacement. This enables aerodynamic optimisation cases that need to preserve the relative shape definition, thus, only allowing the translation of the shape. Figure 3 shows such a

translation, where the relative shape is well preserved with only 2 control nodes. Applications benefiting from this behaviour include optimisation of configurations such as the optimum mounting location of an engine below the wing [41] or angle of attack optimisation of a wing [42]. The second case is conducted as a 2D example in section 3.1.2.

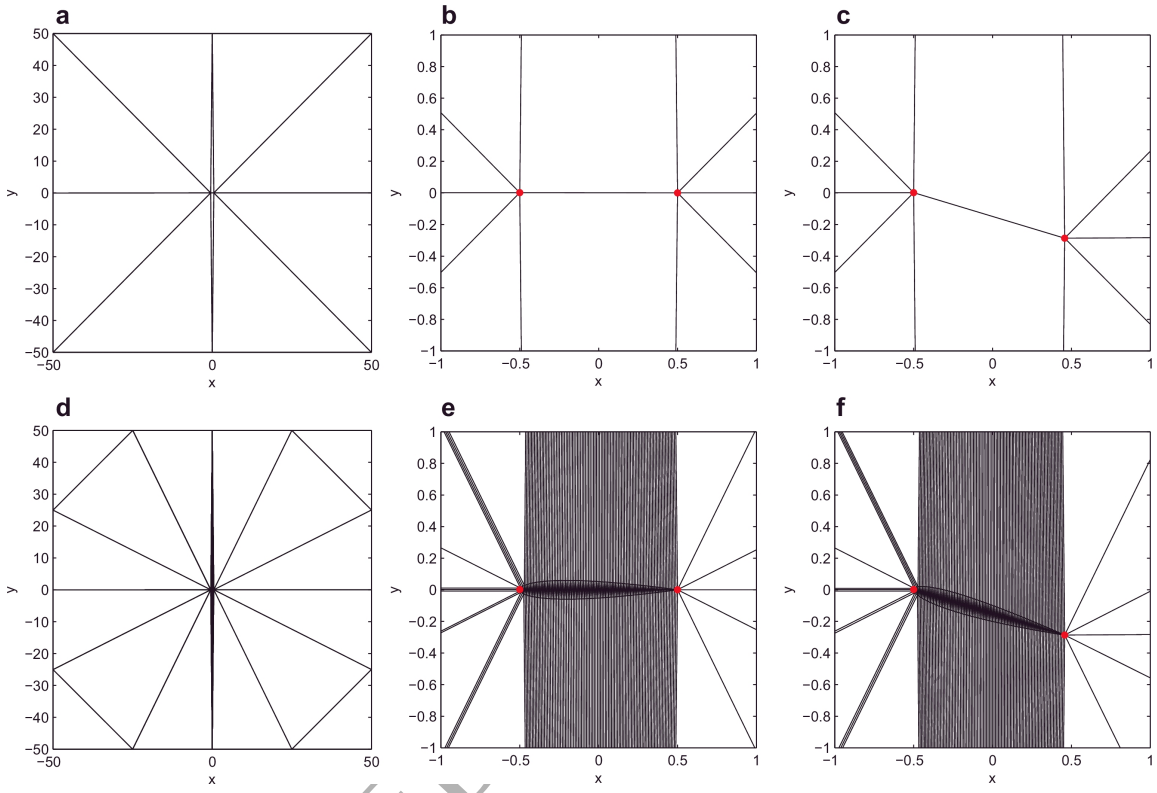


Figure 4: Delaunay Graphs: (a) hyper-coarse DG (b) close-up of (a) (c) Moved hyper-coarse DG (d) Coarse DG (e) close-up of (b) (f) Moved coarse DG Red points - control nodes



### *2.3. Optimisation Approach*

#### *2.3.1. Background*

A significant advantage of the intuitive parameterisation approach is the easy coupling with an optimisation algorithm since the 'control nodes' act naturally as the design parameters during the optimisation process. The components of the position vector of each control node effectively become the parameters of the design space. The large range of control node displacement that are allowed led to the testing of global optimisers in this study, in particular, evolutionary algorithms. Traditionally gradient based optimisers [22, 24, 43] have been the preferred choice for aerodynamic shape optimisation. The large design space exploration possible using the approach identified here, however, always results in the risk of the optimiser 'getting trapped' in local minima.

Genetic Algorithms describe a subclass of such gradient free, evolutionary methods and are heuristic in nature, operate globally and use large populations of agents to search the design space. All Genetic Algorithms follow a similar principle that adopts the idea of 'survival of the fittest' using a series of mutations and crossovers per

optimisation iteration (or 'generation'). In MCS random mutations are achieved using Levy flights. Crossovers are performed by applying the golden ratio rule between two optimal agents in the design space. As the number of generations increases, the best agents ideally approach the global optimum. In contrast to gradient based techniques, testing for global convergence is not straightforward and these types of algorithms tend to be left to run for a pre-set wallclock time, or fixed number of generations, rather than a clearly defined convergence target. The fitness of each agent has to be evaluated at every generation. As a result, the computational efficiency of gradient free techniques is often inferior to gradient based methods due to the large number of fitness evaluations [44]. The problem is particularly significant when considering applications where a single fitness evaluation represents a significant computational cost as is the case in shape optimisation. Nevertheless, gradient based methods are highly depend on the initial design [43] which is something that the authors wished to move away from in this implementation.

A large body of work published on the application of gradient free techniques to shape optimisation focuses on attempts to address the issue of computational cost. This is usually addressed by the use of cheap surrogate or meta models which approximate the expensive objective function [45, 46, 47, 48]. These were not implemented in the study presented here as the focus of this work was the effectiveness and applicability of the parameterisation scheme itself. Future work will be conducted into algorithm acceleration using Reduced Order Modelling.

Modified Cuckoo Search (MCS) was specifically developed with the aim of reducing the number of fitness evaluations required to find the global optimum [49]. Since its introduction MCS has been shown to be efficient when compared with other gradient free techniques in real applications [50, 51] which motivates its use in the present context.

#### *2.3.2. Modified Cuckoo Search*

The basis for the MCS is the Cuckoo Search algorithm (CS). Inspired by the reproduction strategy of cuckoos the CS algorithm [52] has been used in a variety of applications [53, 54, 55, 56, 57]. The key component of the strategy, which is mimicked by CS, is the aggressive behaviour of cuckoos, which lay eggs in the nests of other species. If the cuckoo egg differs significantly from the eggs of the host bird, there is the chance that the host may abandon the nest and the eggs. Cuckoo eggs have slowly evolved to prevent this by mimicking the patterns of the eggs belonging to other local species of birds [58].

In the MCS algorithm the agents, called eggs, are each generated to represent a particular set of design parameters. The initial generation is build by sampling the entire design space. The fitness of all eggs is evaluated and finally the eggs are placed in nests. To model the effect in nature of a host bird discovering eggs with poor fitness, a fraction of eggs with the worst fitness are discarded at each generation and new eggs are generated by performing a random Lévy flight [59]. For the retained assembly of 'best' eggs, each best egg is randomly paired with another best egg and a 'cross-breading' between them is performed to create a new egg, which is kept and placed in a new nest only if the fitness outperforms the fitness of its 'parent eggs'. The optimum ratio of best to worst eggs has been found, empirically, to be 1:3 [49]. The process of replacing and creating eggs continues until a stopping criterion is met (often simply a prescribed number of generations).

#### *2.3.3. Initial Sampling*

In the first generation, a set of agents is required for the MCS algorithm to form the initial population with size $N$. Naturally, the initial geometry provided by the user of the algorithm functions as a starting point.

Based on the initial geometry, the initial population is generated by sampling the entire design space (defined by the allowed range of control node displacements) using Latin Hypercube Sampling (LHS) [60]. Each degree of freedom of the system $d$ (dimensions of the design space) is divided into $N$ equal intervals to create $N^d$ cells. Then, one cell is selected at random. Any other cell in the same interval of every dimension is now excluded. This process is repeated $N$ times until each interval contains exactly one selected cell. The ideal population size $N$ is difficult to pre-determine. It is a common convention used by other researchers in the field [19, 50] to set $N = 10d$ as a suitable compromise between convergence and computational cost.

### *2.4. Computational Fluid Dynamics*

All case studies conducted in this work were two–dimensional in space with the purpose to prove the concept, although all the techniques considered (LHS [60], MCS [49], FDGD [20]) have natural three dimensional extensions. This will be covered in future work. Since the algorithm utilizes the CFD solver as a black box, the CFD approach will only be described briefly.

Two dimensional unstructured triangular meshes were generated using the FORTRAN-based Swansea University FLITE CFD system [32]. The advancing layers technique was used to generate the boundary layer mesh with cell heights defined by the user. An isotropic triangular mesh is then generated using the Delaunay technique with point insertion governed by a pre-defined mesh cell size function across the computational domain.

The FLITE CFD system fluid solver is an edge–based, node–centred finite volume discretisation for solution of the compressible Reynolds Averaged Navier–Stokes equations [61]. The turbulence model applied for the viscous example was the two equation Spallart-Allmaras [62]. Standard subsonic and supersonic far-field boundary conditions were selected. For solid surfaces a no slip condition was imposed for viscous cases and for inviscid cases, they were treated as inviscid walls. In case of the engine intake examples the mass flow was fixed on the compressor intake. Its implementation approach for this is outlined in reference [63]. The results of the CFD solver forms the input for the optimisation algorithm to determine the shape 'fitness' value.

### *2.5. Summary: global methodology*

In order to gain an overview of the entire algorithm it is worth considering the problem as a sensitivity analysis [30]. A general CFD–coupled aerodynamic shape optimisation algorithm can be understood as a series of sensitivity derivatives (which in the case of a gradient free approach are never explicitly computed) through which the sensitivity of the solution fitness $f$ varies with the design parameters $\boldsymbol{v}$ as,

$$\frac{\partial f}{\partial \boldsymbol{v}} = \left[\frac{\partial f}{\partial \boldsymbol{R_m}}\right]\left[\frac{\partial \boldsymbol{R_m}}{\partial \boldsymbol{R_b}}\right]\left[\frac{\partial \boldsymbol{R_b}}{\partial \boldsymbol{R_g}}\right]\left[\frac{\partial \boldsymbol{R_g}}{\partial \boldsymbol{v}}\right] \tag{4}$$

where $\boldsymbol{R_m}$ defines the computational mesh used for the CFD analysis, $\boldsymbol{R_b}$ defines the boundary/surface mesh, and $\boldsymbol{R_g}$ is the geometry definition. One of the advantages of the control node parameterisation outlined in this paper is that the control nodes are used both to move the boundary and to parametrically provide a geometry definition such that the third and fourth sensitivity derivatives collapse to become

$$\left[\frac{\partial \boldsymbol{R_b}}{\partial \boldsymbol{R_g}}\right]\left[\frac{\partial \boldsymbol{R_g}}{\partial \boldsymbol{v}}\right] = \left[\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{C}}\right] \tag{5}$$

where $\boldsymbol{v} = \boldsymbol{C}$ and $\boldsymbol{R_b}$ becomes $\boldsymbol{B}$. Equation 2 outlines the simple relationship between $\boldsymbol{B}$ and $\boldsymbol{C}$. Through the implementation of the FDGD method this system further collapses and yields

$$\left[\frac{\partial \boldsymbol{R_m}}{\partial \boldsymbol{B}}\right]\left[\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{C}}\right] = \left[\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{C}}\right] \tag{6}$$

since the effect of control node displacement is directly propagated throughout the entire mesh $\boldsymbol{M}$. Finally, Equation 4 becomes

$$\frac{\partial f}{\partial \boldsymbol{C}} = \left[\frac{\partial f}{\partial \boldsymbol{M}}\right]\left[\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{C}}\right] \tag{7}$$

and sensitivity analysis is reduced to two parts. In the approach outlined in this paper, the sensitivity $\left[\frac{\partial f}{\partial \boldsymbol{M}}\right]$ is determined using the FLITE CFD solver and $\left[\frac{\partial \boldsymbol{M}}{\partial \boldsymbol{C}}\right]$ is never explicitly computed but the population of agents are 'steered' by MCS and the effect of this is propagated to the computational mesh via FDGD.

A flow chart of the overall algorithm is given in figure 5. The entire algorithm has been developed in FORTRAN and the CFD solver is embedded as a black box. The code has been parallelised to take advantage of High Performance Computing with each agent in the population being allocated a PC cluster core.

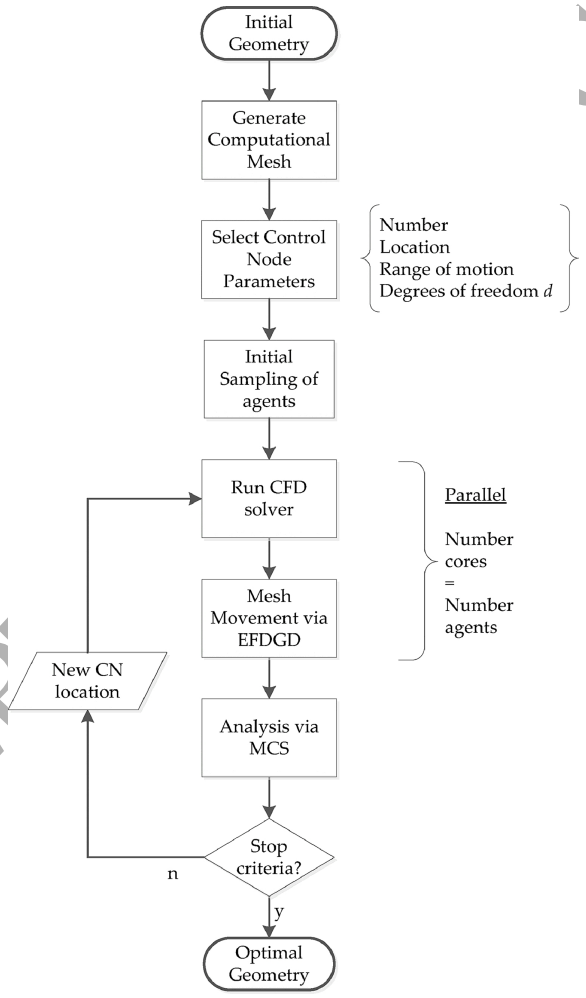


Figure 5: Flow chart of the overall algorithm developed in this paper

## 3. Case Studies

Four 2D aerodynamic design case studies, of varying complexity, are considered in this section to demonstrate the characteristics of the algorithm presented. The first two cases are simple angle of attack optimisation cases to illustrate the effectiveness of the algorithm regarding shape preservation as outlined in section 2.2.3. Finally, the algorithm is applied to the problem of subsonic, transonic and supersonic engine intake duct optimisation in order to test its robustness and effectiveness regarding shape optimisation. All case studies have been run on the HPC Wales machine [64], a super-cluster exhibiting Westmere/Sandy Bridge processors with approximately 2.6 Ghz per core. Any computation, processing and run times provided in this section are based on these processors.

The population size within each case study was calculated according to the $N = 10d$ rule of thumb. Furthermore, a generous stopping criteria of 100 Cuckoo generations has been set for all cases to achieve a comprehensive understanding of the algorithm functionality in the context of convergence behaviour.

### *3.1. Aerofoil*

Two simple aerofoil configuration optimisation cases are first considered in which the preservation of the aerofoil shape is desired but the angle of attack of the aerofoil is varied. Both utilise the NACA 0012 aerofoil. Flow was considered to be inviscid and at a Mach number of 0.5. The mesh is shown in Figure 6 and contains 4551 nodes and 8884 triangular elements.

Two control nodes were placed at the leading edge and trailing edge to control the aerofoil angle of attack. The first control node at the leading edge $C_1(x_{cn_1}, y_{cn_1})$ was fixed and the second control node at the trailing edge $C_2(x_{cn_2}, y_{cn_2})$ was constrained to an arc defined as

$$(x_{cn_2} - x_{cn_1})^2 + (y_{cn_2} - y_{cn_1})^2 = c^2 \tag{8}$$

where $c$ is the chord length of the aerofoil. $C_2$ travels a distance $s$ along the arc dependent on the change in angle of attack $\Delta\alpha$, which is given by

$$s = \frac{\pi}{180}\Delta\alpha c \tag{9}$$

As a result, the problem was reduced to one degree of freedom ($d = 1$), as the design parameter is solely the position of the control node on the arc distance $s$. The explorable design space was limited by constraining the range of motion of control node $C_2$ to $s \in [-\frac{\pi}{18}, \frac{\pi}{18}]$. Note that Table 1 summarizes all input parameters and results for the full range of case studies considered.

#### *3.1.1. Fitness (objective function)*

The fitness $f_1$ for the first case was,

$$f_1 = -\left|L\right|/q_\infty \tag{10}$$

and the fitness $f_2$ for the second case was,

$$f_2 = \frac{L}{D} \tag{11}$$

where $q_\infty$ is the freestream dynamic pressure, $L$ is aerofoil lift and $D$ is aerofoil drag defined as,

$$L = \oint p(\boldsymbol{n}.\boldsymbol{j})\,\mathrm{d}B_a \tag{12}$$

$$D = \oint p(\boldsymbol{n}.\boldsymbol{i})\,\mathrm{d}B_a \tag{13}$$

Here, $p$ is the non-dimensionalized static pressure, $\boldsymbol{n}$ is the normal unit vector directing into the surface, $\boldsymbol{i}$ and $\boldsymbol{j}$ are the parallel and vertical unit vectors in relation to the freestream velocity direction and $B_a$ is the aerofoil shape boundary.

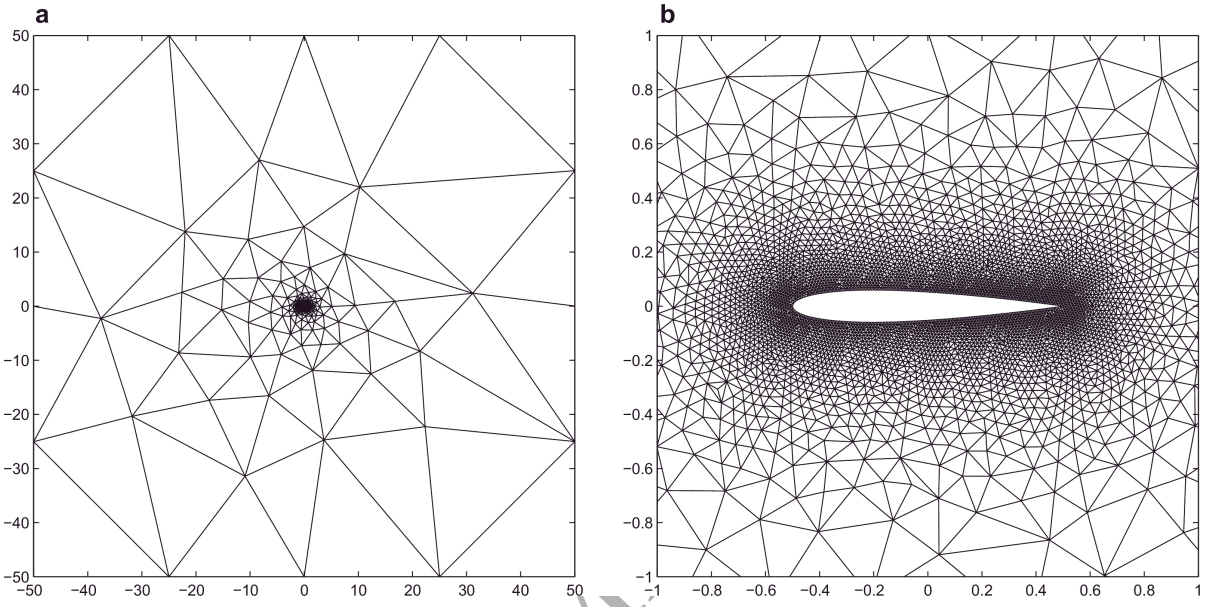


Figure 6: Initial aerofoil mesh (a) entire domain (b) zoom

*3.1.2. Lift minimisation*

The first case was a very basic proof of concept in which the angle of attack $\alpha$ of an aerofoil was optimised for minimal lift as given in equation 10. Here, the optimal solution is known to be $\alpha = 0$, since NACA0012 is a symmetrical aerofoil. Hence, the final outcome can be compared against the known global optimum. The initial geometry was created at an angle of $\alpha = 9°$. The results in figure 7 show a rapid convergence towards the known optimum within 10 generations. Note, that the improvement seen in the initial generation is a result of the 'brute force' sampling rather than the Cuckoo search optimisation. The sampling was included in the figure being the first generation of the optimisation. The subsequent convergence displayed after generation one in the close-up of Figure 7 (a) was achieved by the MCS. Afterwards, slight changes occur for example at step 29. Nonetheless, the MCS never reaches the final optimum with a final fitness of -0.06 $m^2$ over an initial value of -0.89 $m^2$.

The total computational expense for the given set-up and 10 generations is approximately 9.5 core hours, which is unreasonable for a simple case as presented here (which could easily have been solved using a gradient based optimiser). Nevertheless, this case demonstrated that the optimiser and mesh movement algorithm were both working effectively.

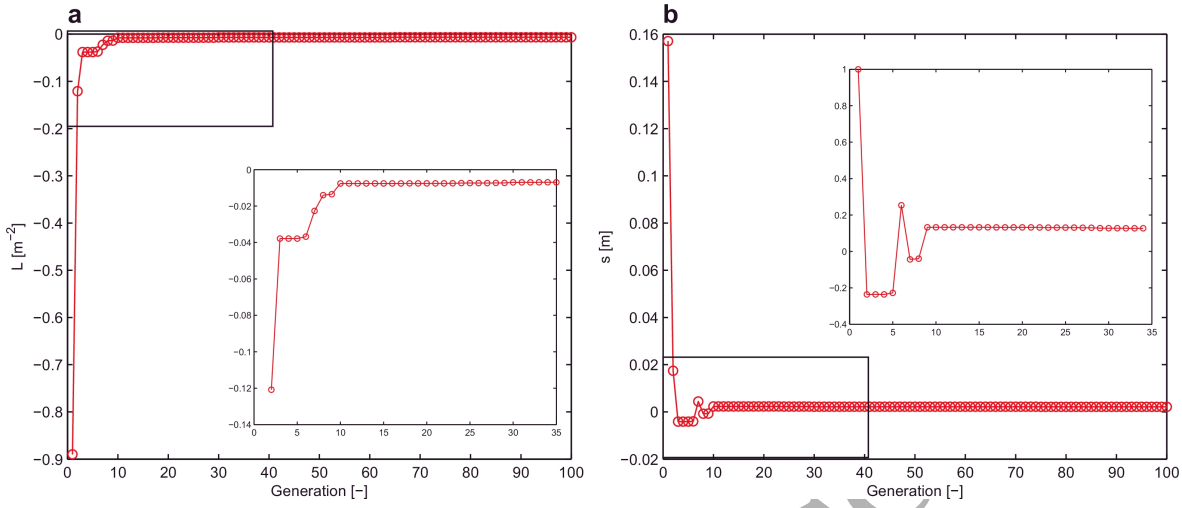


Figure 7: Fitness results of minimising for Lift including close-ups (a) Lift development over all cuckoo generations (b) change of trailing edge control node $C_2$ position over cuckoo generations

### *3.1.3. Lift to Drag maximisation*

The second case is another example of an optimisation requiring shape preservation in which the angle of attack $\alpha$ of an aerofoil was optimised for maximizing L/D according to equation 11. The level of shape preservation is illustrated in figure 8 for an example case of $s = \frac{\pi}{18}$ showing negligible discrepancy between the starting shape and the moved shape at its optimum angle of attack. An initial angle of $\alpha = 0°$ was defined

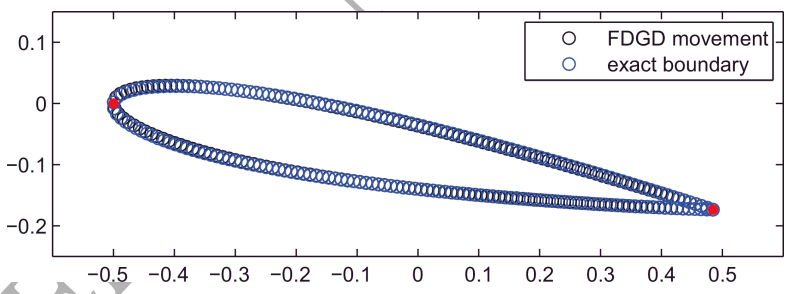


Figure 8: MODIFIED Boundary shape after moving $C_2$ from $\alpha = 0°$ for $s = \frac{\pi}{18}$ using FDGD in comparison with exact boundary

and the movement of the control node $C_2$ at the trailing edge was again constrained providing the relationship in equation 9. This time, the optimal solution was less obvious compared to the previous case and has been estimated based on the data received from every agent during the run. The result is plotted in Figure 9 presenting the fitness of every agent (L/D) over the respective control node positions. In comparison, Figure 10 illustrates the convergence of the fitness and the change in control node position over the generations using MCS. It can be observed, that a solution in the area of the global optimum was reached taking noise effects in the fitness evaluation into account. However it is unclear, whether the final optimum has been obtained sufficiently. The final fitness value (L/D) is 32.4 and 6.9 % of the improvement was achieved after the initial sampling.

When compared to the previous case, a fast convergence in the initial phase becomes apparent with almost no changes appearing after generation 10. A comparison of the initial and final pressure field in Figure 11 illustrates, that the pressure distribution has significantly improved. The computational expense is equal with approximately 9.5 h per 10 generations.

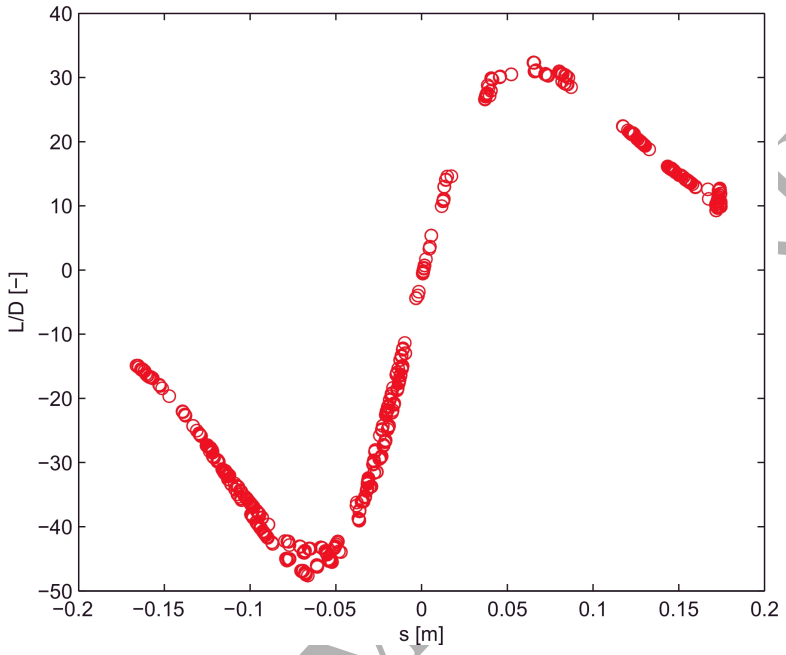


Figure 9: Lift to Drag over the control node position

### *3.2. Intake Duct Optimisation*

Having validated the algorithm's ability to find known optima, a more sophisticated aerodynamic optimisation problem was tackled. A common problem definition in aerodynamic design is the optimisation of a jet engine intake duct[5] tailored to a specific aircraft configuration. In order to ensure efficient performance and avoid stall or surge, the flow pattern at the engine's compressor face must be satisfactory across the entire speed range and flight conditions that the vehicle will experience. Experience and engineering intuition are often the primary tools used to solve this aerodynamic design problem in industry[63, 32].

The starting geometry was an engine intake duct of a land–based supersonic vehicle. The mesh utilized is illustrated in Figure 12 and exhibits 82868 mesh nodes and 163419 mesh elements. Solutions to the problem were sought at a range of Mach numbers $Ma = [0.5, 0.8, 1.1, 1.4]$ using viscous CFD simulations. To capture viscosity effects, a triangulated boundary layer mesh with 7 layers was integrated into the mesh. The Y+ value 1.0 for the first layer and mesh layers growing exponentially to meet the background volume mesh size by layer 7 were chosen. No wall functions were used. Note that, in this optimisation concept proof study, achieving highly accurate CFD simulations was deemed to be less important than developing an understanding of the optimiser characteristics. The number of control nodes differed between both cases. At first, one control node has been defined at the tip of the upper lip of the intake. Afterwards, an additional three control nodes were added.

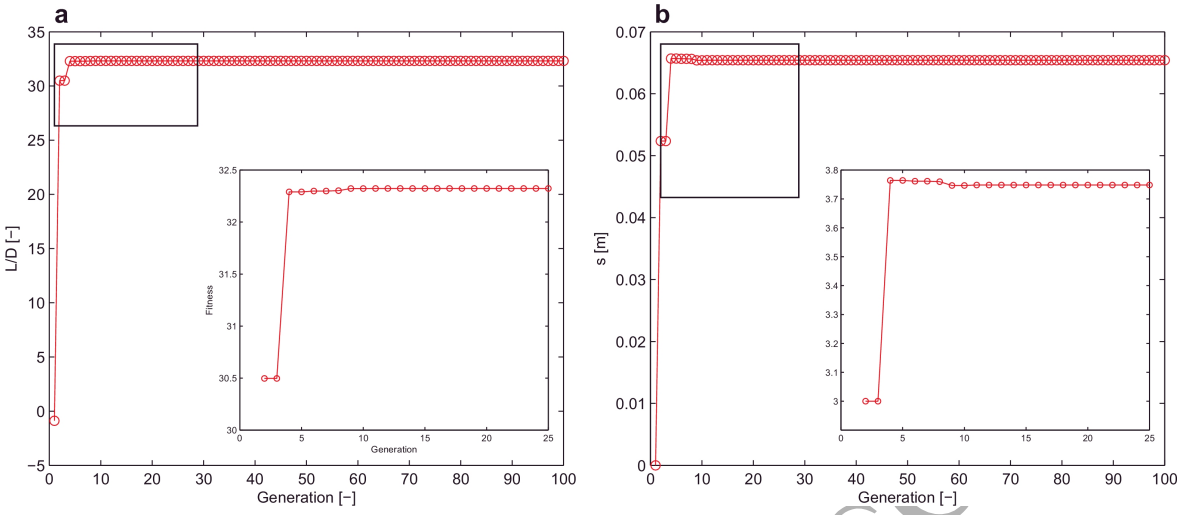


Figure 10: Fitness results of maximising Lift to Drag ratio including close-ups (a) Lift to Drag development over all cuckoo generations (b) change of trailing edge control node $C_2$ position over cuckoo generations

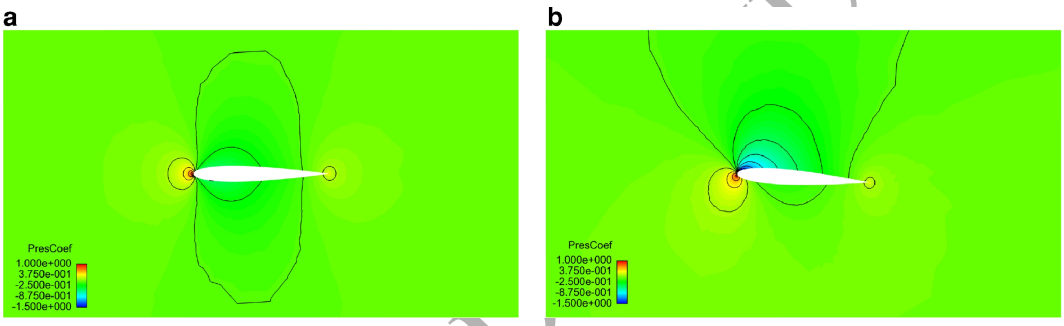


Figure 11: Field of pressure coefficients of aerofoil (a) initial (b) final

Figure 12 (b) visualises their location. All control nodes were allowed movements in both vertical and horizontal direction with a specified explorable design space of $x_C \in [-0.3, 0.3]$ and $y_C \in [-0.3, 0.3]$ for each control node $C$, where the design space has been normalized in relation to $L$ of equation 14. Table 1 summarizes all parameters.

*3.2.1. Fitness (objective function)*

Distortion is one of the main flow parameters a jet engine intake is designed for and provides a measure of standard deviation of the total pressure across a plane of interest, in this case the jet engine compressor face. The deviation across the plane should be kept minimal for an optimum flow pattern into the engine. Fitness is always defined in the context of maximization. Thus, in order to minimize distortion, fitness is a product of the negative distortion $\sigma$.

$$F = -\sigma = -\int_0^L \frac{|P_t - \bar{P}_t|}{\bar{P}_t L}\,\mathrm{d}l \tag{14}$$

where $P_t$ is the total pressure and $\bar{P}_t$ is the mean total pressure. $l$ is a coordinate moving along a line in 2D, both of which are defined along the engine inlet. The equation is normalized against $L$, which is the length of

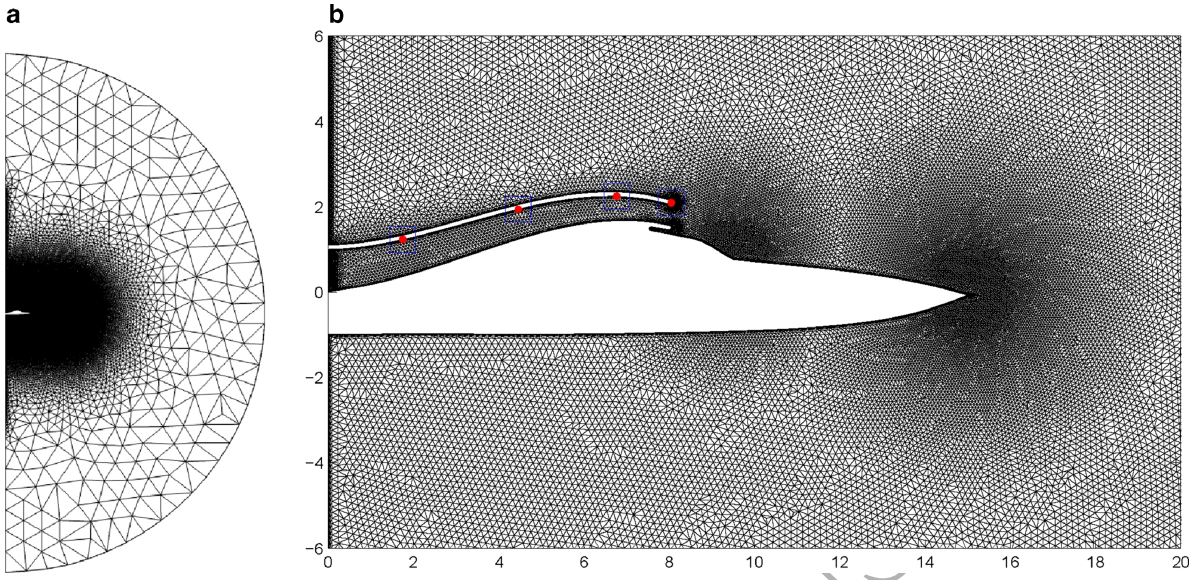


Figure 12: Mesh of supersonic vehicle with engine intake duct (a) total mesh (b) detailed mesh including the control node locations and their range of motion

the engine inlet.

*3.2.2. 1 Control Node case*

For a third case, one control node has been selected with allowed movements in both dimension x and y. Therefore, the population size was set 20. However, since the calculation of the optimal population size is a subjective decision as outlined in section 2.3.3, the case has been utilized to test the effect of varying population sizes. Thus, the same case was performed twice with a population size of 10 and 20. An overview of the fitness development of both cases for all Mach numbers $Ma = [0.5, 0.8, 1.1, 1.4]$ are shown below in Figure 13 and all fitness values are detailed in table 1.

As a general observation, all fitness values have improved over the Cuckoo generations. This becomes more evident in Figure 14, that visualises the flow patterns across all Mach numbers for the case with a population size of 10 emphasizing the engine inlet. A more homogeneous pressure pattern was obtained at the engine inlet resulting in a reduction in pressure distortion (fitness). Additionally, in Figure 13, an expected improvement of the final results can be seen for the case with increased population. An average improvement across all four Mach numbers of 6.65 % has been achieved illustrating the benefit of an increased population for otherwise equal input parameters. It can be argued, that a twofold increase in number of generations possibly has a similar impact. However, in the context of wall-clock time efficiency for a parallelised code, an increase in the number of agents is more advantageous over increasing the number of generations.

When further comparing both cases with different population size no sign of faster convergence towards an optimum can be detected as both runs show examples with partly significant changes close to the end. Besides, large jumps in the fitness may occur even after a period of almost no change in fitness and a perceived convergence. This outlines the difficulty to define a suitable convergence criteria or population size for a generic engineering problem with unknown outcome when using global optimisation algorithms. Most certainly, population size

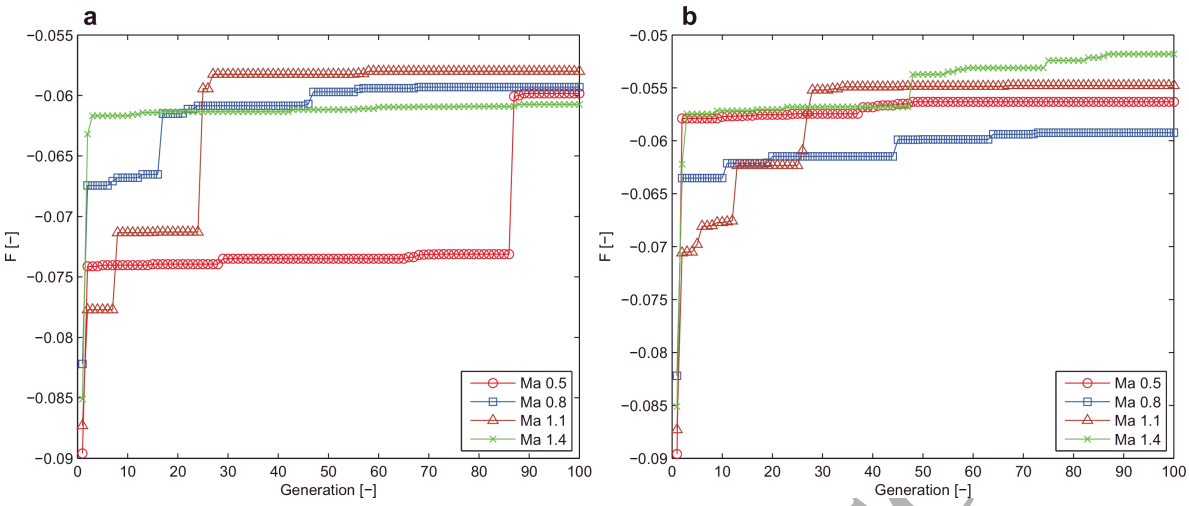


Figure 13: MODIFIED Fitness development over cuckoo generations with population size of (a) 10 (b) 20

and number of generations should therefore be driven by the real time and computation time constraints of an individual project.

Figure 15 displays the movements of the control node of the agent with the best fitness at each generation. The results coincide with the characteristics in Figure 13. Active control nodes generate better results compared to control nodes with little motion. Only the large jump in fitness of the Ma 0.5 run with a population size of 10 is not reflected in the motion of its control node. In conclusion, the activity of the control node may well be an indicator of convergence towards a global optimum, however, a sudden discovery of a new global optimum is always possible.

All four Mach numbers for a population size of 20 required approximately 10,000 core hours and 5,000 core hours for a population size of 10.

### *3.2.3. 4 Control Nodes case*

Finally, four control nodes have been applied with allowed movements in both dimension as prescribed in Figure 12 (b). Again the case was conducted for four Mach numbers. The population size was set to 80. The fitness development of all cases is shown in Figure 16.

Again, all fitness values have improved over the Cuckoo generations. Table 1 provides details of all values including start and end values of the fitness. The improvement is more apparent in Figure 17, that visualises the flow pattern for Mach number 1.4 emphasizing the engine inlet. A more homogeneous pressure pattern was obtained at the engine inlet resulting in a reduction in pressure distortion (fitness). It needs to be noted, that the rather unusual optimised shape is a result of the choice of objective function. A multi-objective optimisation including other important factors in the design (for example total flow rate) would most likely result in a different shape but this is beyond the purpose of this paper which is simply to prove the concept. In Figure 16, the convergence pattern over the generations is comparable to the previous case. After the first significant step generated by the initial sampling, the fitness advances further due to the MCS until only little changes occur. Nonetheless, the level of improvement has increased by a magnitude of approximately 3 due to more flexibility by implementing three more control nodes. In accordance with the latter case, the majority of improvement is

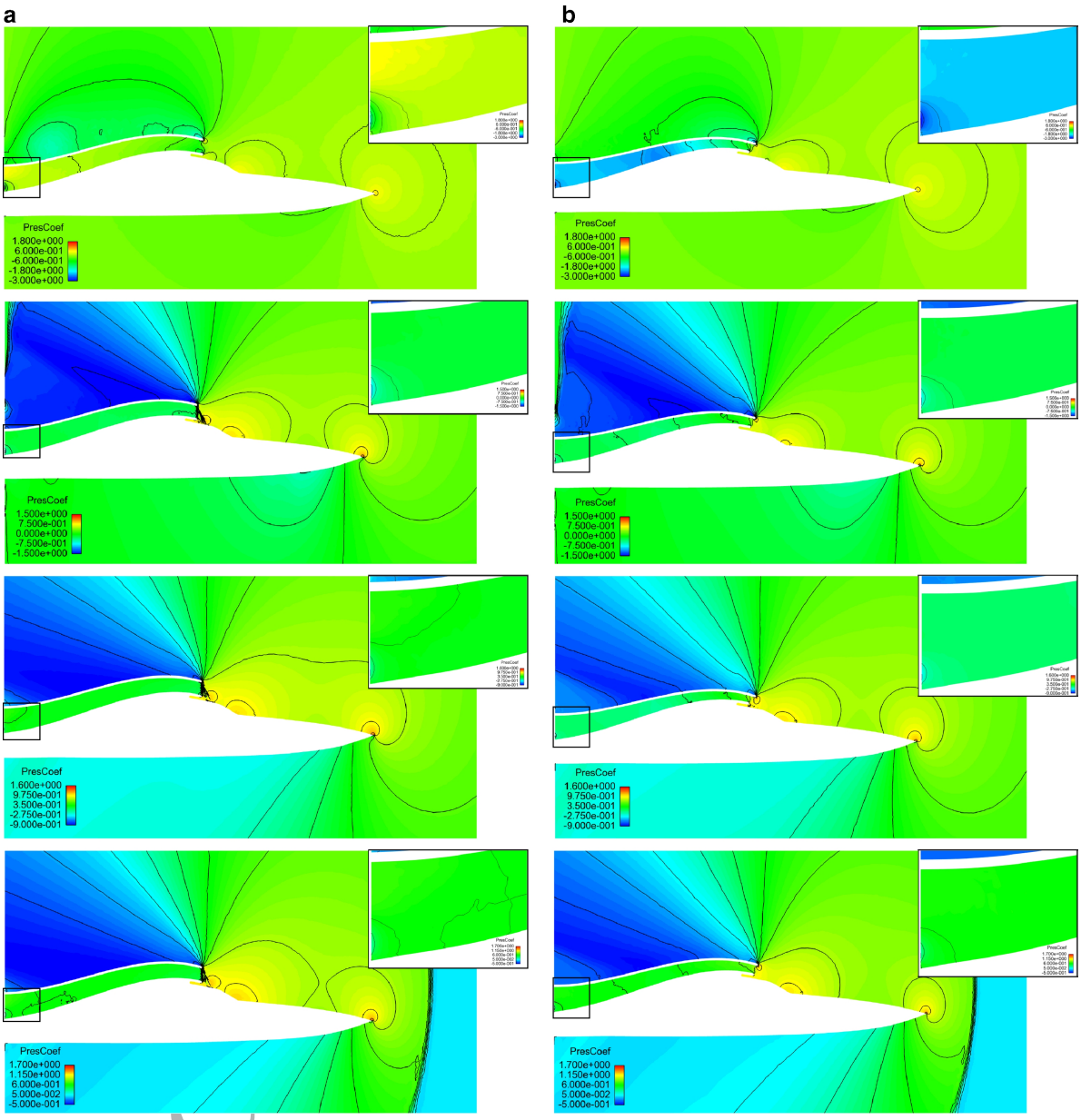


Figure 14: Pressure Fields of the supersonic vehicle with engine intake duct of (a) initial geometry (b) optimised geometry for Ma 0.5, 0.8, 1.1, 1.4

achieved in the early stages. Looking at the behaviour of the control nodes in Figure 18, one may notice the large magnitude of change in control node position occurring for most Mach numbers. That implies a more complex design space with an increasing number of local optima existing in the objective function. It leads to the conclusion, that the MCS algorithm is very good at finding global optima in design space but struggles in converging precisely to the local optima.

Finally, the effect of control nodes appear to differ, since the control node 3 changes less particularly in the

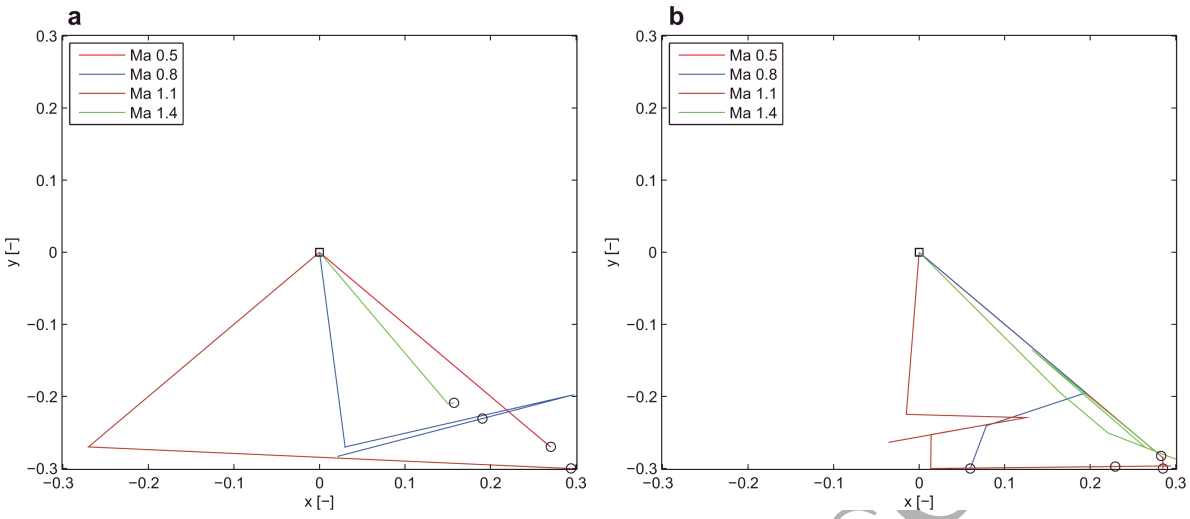


Figure 15: Control Node movement of best egg with population size of (a) 10 (b) 20 □ - start point o - end point

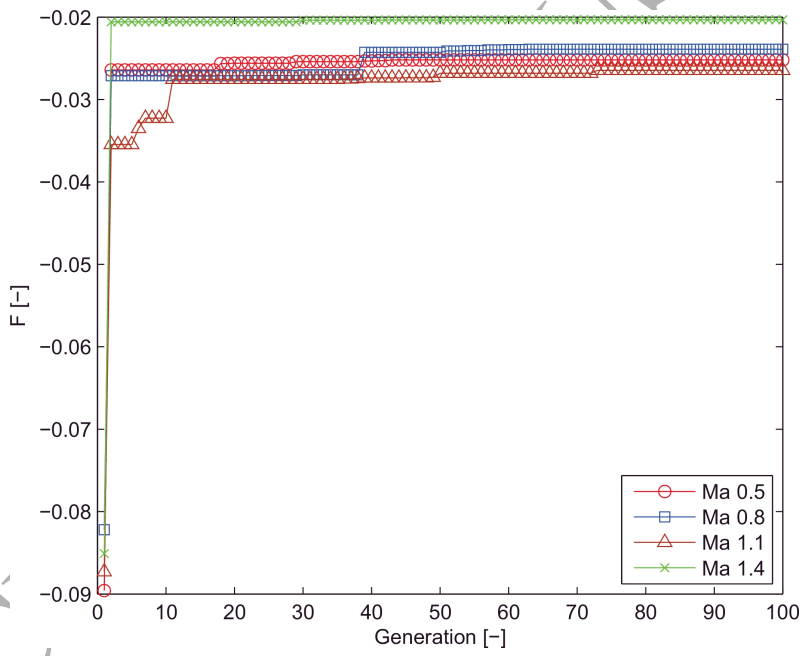


Figure 16: MODIFIED Fitness development over cuckoo generations of 4 CN case

vertical motion compared to other nodes. That raises the question whether control nodes may be more effective at some locations than on others.

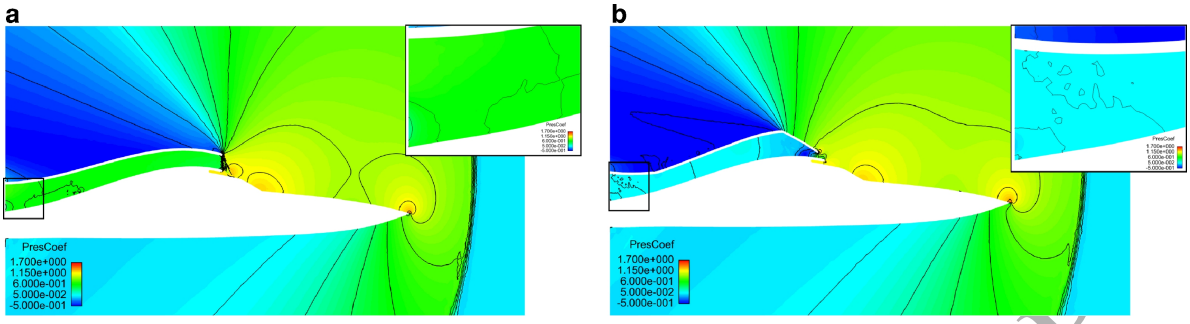


Figure 17: Pressure Fields of (a) initial geometry (b) optimised geometry for Ma 1.4

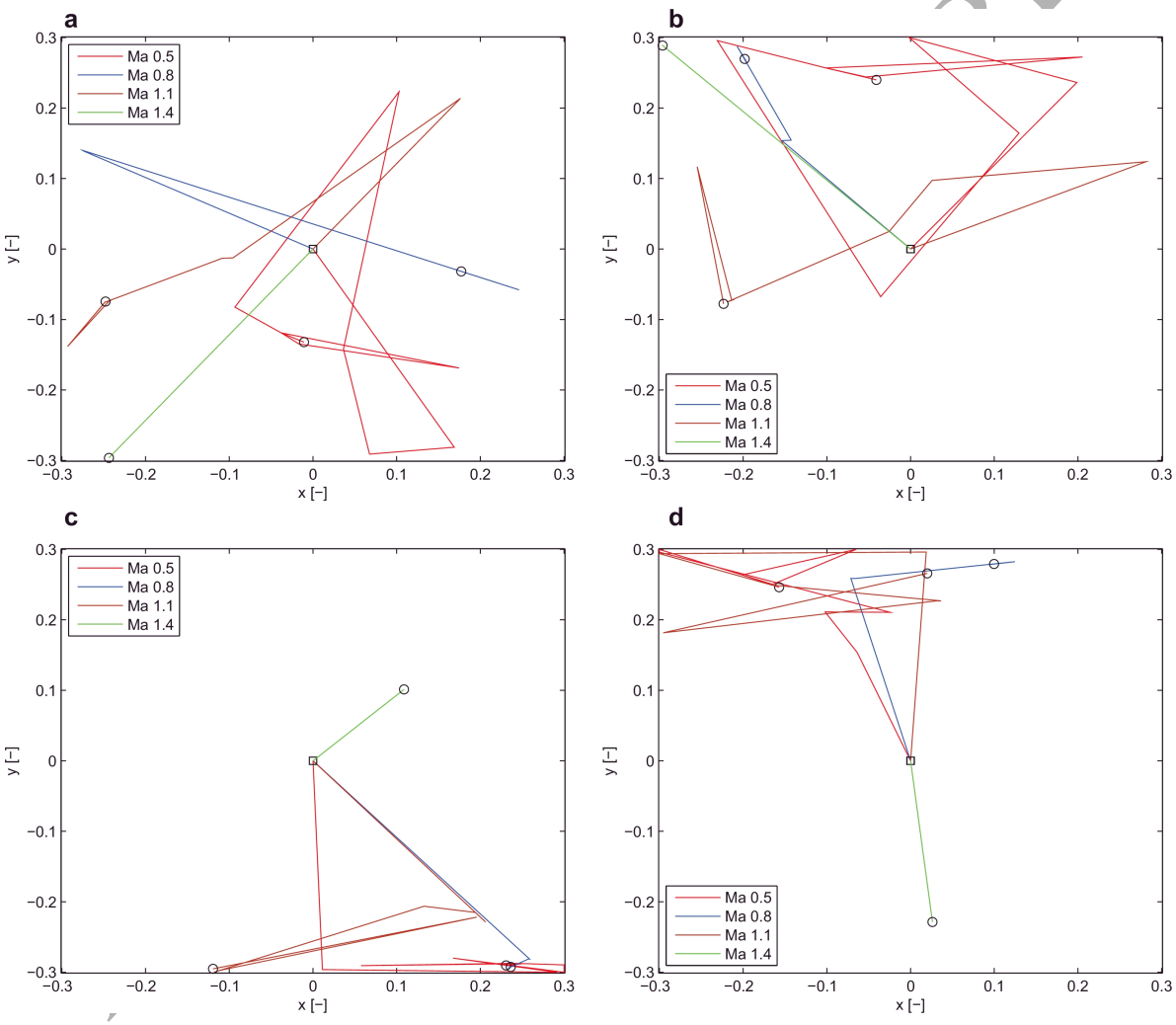


Figure 18: Control node movements of best egg starting from their initial position (a) $C_1$ (b) $C_2$ (c) $C_3$ (d) $C_4$ □ - start point o - end point

| Case | Mesh | | Control Nodes | | Agents | Ma | Re | Fitness | | | CPU | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $n_p$ | $n_e$ | $n_{cn}$ | d | | | | Start | After LHS | End | h | Cores |
| 3.1.2 | 4551 | 8884 | 2 | 1 | 10 | 0.5 | $\infty$ | -0.89 | -0.12 | -0.06 | 95 | 10 |
| 3.1.3 | 4551 | 8884 | 2 | 1 | 10 | 0.5 | $\infty$ | 0.03 | 30.5 | 32.4 | 95 | 10 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 10 | 0.5 | 6.5e5 | -0.089 | -0.074 | -0.060 | 5e3 | 10 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 10 | 0.8 | 6.5e5 | -0.082 | -0.068 | -0.059 | 5e3 | 10 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 10 | 1.1 | 6.5e5 | -0.087 | -0.078 | -0.058 | 4.5e3 | 10 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 10 | 1.4 | 6.5e5 | -0.085 | -0.063 | -0.061 | 4e3 | 10 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 20 | 0.5 | 6.5e5 | -0.089 | -0.058 | -0.056 | 10e3 | 20 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 20 | 0.8 | 6.5e5 | -0.082 | -0.064 | -0.059 | 10e3 | 20 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 20 | 1.1 | 6.5e5 | -0.087 | -0.071 | -0.055 | 9e3 | 20 |
| 3.2.2 | 82868 | 163419 | 1 | 2 | 20 | 1.4 | 6.5e5 | -0.085 | -0.062 | -0.052 | 8e3 | 20 |
| 3.2.3 | 82868 | 163419 | 4 | 8 | 80 | 0.5 | 6.5e5 | -0.089 | -0.027 | -0.025 | 40e3 | 80 |
| 3.2.3 | 82868 | 163419 | 4 | 8 | 80 | 0.8 | 6.5e5 | -0.082 | -0.027 | -0.024 | 40e3 | 80 |
| 3.2.3 | 82868 | 163419 | 4 | 8 | 80 | 1.1 | 6.5e5 | -0.087 | -0.035 | -0.026 | 36e3 | 80 |
| 3.2.3 | 82868 | 163419 | 4 | 8 | 80 | 1.4 | 6.5e5 | -0.085 | -0.021 | -0.020 | 32e3 | 80 |

Table 1: Summary table of all input parameters. The first two cases are aerofoil cases of section 3.1. All following cases display engine intake duct optimisation cases of section 3.2, that differ in their Mach number or number of agents.

## 4. Conclusions

An new computational aerodynamic shape optimisation algorithm has been developed making use of the novel concept of control nodes. The control nodes, located on the discrete shape boundary, are used for both defining the geometry movements and as the design parameters for the optimisation process. The approach has been coupled to the FDGD mesh movement technique to propagate the displacement of the control nodes, driven by a Modified Cuckoo Search evolutionary optimiser to the computational mesh for CFD analysis. The approach is CAD-free and requires no remeshing between generations. The resulting algorithm has been successfully applied to four different aerodynamic case studies including aerofoil lift minimisation, lift–drag ratio optimisation optimisation and subsonic, transonic and supersonic engine intake duct design. It is demonstrated to be effective across a range of aerodynamic design problems, is intuitive for the user and significant improvements in the fitness were achieved over relatively few generations. It also benefits from the property of shape preservation where it necessary simply to optimise based on translation or rotation of parts of the geometry. In future work, the algorithm is being developed to include a CFD acceleration approach such as Reduced Order Modelling, optimised and variable control node location selection and extension to three dimensions.

## Acknowledgements

The authors would like to acknowledge the support provided by Fujitsu and the HPC Wales for provision of PhD studentship funding and computational support.

## References

[1] O. Hassan, K. Morgan, E. J. Probert, and J. Peraire, "Unstructured tetrahedral mesh generation for three–dimensional viscous flows," *International Journal for Numererical Methods in Engineering*, vol. 39, pp. 549–567, 1996.

[2] N. P. Weatherill and O. Hassan, "Efficient three–dimensional delaunay triangulation with automatic boundary point creation and imposed boundary constraints," *International Journal for Numererical Methods in Engineering*, vol. 37, pp. 2005–2039, 1994.

[3] K. Morgan and J. Peraire, "Unstructured grid finite element methods for fluid mechanics," *Reports on Progress in Physics*, vol. 61, pp. 569–638, 1998.

[4] N. P. Weatherill and C. R. Forsey, "Grid generation and flow calculation for complex aircraft geometries using a multi–block scheme," *AIAA Paper*, vol. 85, 1985.

[5] S. Allright, "Multiblock topology specification and grid generation for complete aircraft configurations," *Applications of Mesh Generation to Complex 3D Configurations (Conf. Proc. No 464)*, pp. 11.1–11.11, 1989.

[6] A. J. Keane and P. B. Nair, *Computational Approaches for Aerospace Design*. Wiley, June 2005.

[7] A. Jameson and J. C. Vassberg, "Computational fluid dynamics: Its current and future impact," *AIAA Paper 2001-0538*, 2001.

[8] S. Shahpar, "Challenges to overcome for routine usage of automatic optimisation in the propulsion industry," *The Aeronautical Journal*, vol. 115, no. 1172, 2011.

[9] T. Hughes, J. Cottrell, and Y. Bazilevs, "Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement," *Computational Methods of Applied Mechanical Engineering*, vol. 194, no. 1, pp. 4135–4195, 2005.

[10] B.-U. Park, Y.-D. Seo, O. Sigmund, and S.-K. Youn, "Shape optimization of the stokes flow problem based on isogeometric analysis," *Structural Multidisciplinary Optimisation*, vol. 48, no. 1, pp. 965–977, 2013.

[11] P. Nortoft and J. Gravesen, "Isogeometric shape optimization in fluid mechanics," *Structural Multidisciplinary Optimisation*, vol. 48, no. 1, pp. 909–925, 2013.

[12] R. Sevilla, S. Fernández-Méndez, and A. Huerta, "NURBS-enhanced finite element method (NEFEM): A seamless bridge between CAD and FEM," *Archives of Computational Methods in Engineering*, vol. 18, no. 4, pp. 441–484, 2011.

[13] R. Sevilla, S. Fernández-Méndez, and A. Huerta, "3D-NURBS-enhanced finite element method (NEFEM)," *International Journal for Numerical Methods in Engineering*, vol. 88, no. 2, pp. 103–125, 2011.

[14] M. J. Mifsud and D. G. M. S. T. Shaw, "A high fidelity low–cost aerodynamic model using proper orthogonal decomposition," *International Journal for Numerical Methods in Fluids*, vol. 63, pp. 468–494, 2010.

[15] T. Lieu, C. Farhat, and M. Lesoinne, "Reduced–order fluid–structure modeling of a complete aircraft configuration," *Computational Methods of Applied Mechanical Engineering*, vol. 195, pp. 5730–5742, 2006.

[16] N. Queipo, R. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Tucker, "Surrogate-based analysis and optimization," *Progress in Aerospace Science*, vol. 41, no. 1, pp. 1–28, 2005.

[17] A. Forrester and A. Keane, "Recent advances in surrogate-based optimization," *Progress in Aerospace Science*, vol. 45, no. 1-3, pp. 50–79, 2009.

[18] E. Arian and A. Iolo, *Computational Fluid Dynamics Review*. World Scientific, 2010.

[19] S. Walton, O. Hassan, and K. Morgan, "Selected engineering applications of gradient free optimisation using cuckoo search and proper orthogonal decomposition," *Archives of Computational Methods in Engineering*, vol. 20, pp. 123–154, 2013.

[20] X. Liu, N. Qin, and H. Xia, "Fast dynamic grid deformation based on delaunay graph mapping," *Journal of Computational Physics*, vol. 211, pp. 405–423, 2006.

[21] A. Jameson, "Aerodynamic design via control theory," *Journal of Scientific Computing*, vol. 3, pp. 233–269, 1988.

[22] A. Jameson, ed., *Efficient Aerodynamic Shape Optimization*, 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2004.

[23] J. Vassberg and A. Jameson, "Aerodynamic shape optimization of a reno race plane," *International Journal of Vehicle Design*, vol. 28, no. 4, pp. 318–338, 2002.

[24] S. Xu, W. Jahn, and J.-D. Mller, "Cad-based shape optimisation with cfd using a discrete adjoint," *International Journal for Numerical Methods in Fluids*, 2013.

[25] R. Picket, M. Rubinstein, and R. Nelson, "Automated structural synthesis using a reduced number of design coordinates," *AIAA Journal*, vol. 11, no. 4, pp. 4994–498, 1973.

[26] A. Watt and M. Watt, *Advanced animation and rendering techniques*. Addison-Wesley, 1992.

[27] S. S. Sarakinos, E. Amoiralis, and I. K. Nikolos, eds., *Exploring Freeform Deformation Capabilities in Aerodynamic Shape Parameterization*, EUROCON, November 2005.

[28] C. Allen and T. Rendall, "Aerodynamic shape optimisation of hovering rotors using compressible cfd," *The Aeronautical Journal*, vol. 115, no. 1170, pp. 513–519, 2011.

[29] W. Anderson, S. Karman, and C. Burdyshaw, "Geometry parameterisation method for multidisciplinary applications," *AIAA Journal*, vol. 17, no. 6, pp. 1658–1578, 2009.

[30] J. Samareh, "A review of shape parameterization techniques," *NASA*, vol. 209136, pp. 333–343, 1999.

[31] B. J. e. a. Evans, "Simulating steady state and transient aerodynamic flows using unstructured meshes and parallel computers," *World Scientific Review*, 2013.

[32] B. Evans, T. Morton, L. Sheridan, O. Hassan, K. Morgan, J. W. Jones, M. Chapman, R. Ayers, and I. Niven, "Design optimisation using computational fluid dynamics applied to a landbased supersonic vehicle, the bloodhound ssc," *Structural Multidisciplinary Optimization*, vol. 47, pp. 301–316, 2013.

[33] A. de Boer, M. van der Schoot, and H. Bijl, "Mesh deformation based on radial basis function interpolation," *Computers and Structures*, vol. 85, pp. 784–795, 2007.

[34] J. T. Batina, "Unsteady euler algorithm with unstructured dynamic mesh for complex–aircraft aeroelastic analysis," *Tech Rep AIAA–89–1189*, 1989.

[35] C. Farhat, C. Degrand, B. Koobus, and M. Lesoinne, "Torsional springs for two-dimensional dynamic unstructured fluid meshes," *Computational Methods of Applied Mechanical Engineering*, vol. 163, pp. 231–245, 1998.

[36] C. Degand and C. Farhat, "A three–dimensional torsional spring analogy method for unstructured dynamic meshes," *Computational Structure*, vol. 80, pp. 305–316, 2002.

[37] D. Lynch and K. ONeill, "Elastic grid deformation for moving boundary problems in two space dimensions," *Finite elements in water resources*, 1980.

[38] M. Potsdam and G. Guruswamy, "A parallel multiblock mesh movement scheme for complex aeroelastic applications," *Tech Rep AIAA–2001–0716*, 2001.

[39] T. C. S. Rendall and C. B. Allen, "Efficient mesh motion using radial basis functions with data reduction algorithms," *Journal of Computational Physics*, vol. 228, pp. 6231–6249, 2009.

[40] S. Jakobsson and O. Amoignon, "Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization," *Computers and Fluids*, vol. 36, pp. 1119–1136, 2007.

[41] L. Jing, G. Zhenghong, H. Jiangtao, and Z. Ke, "Aerodynamic design optimization of nacelle/pylon position on an aircraft," *International Forum on Aeroelasticity and Structural Dynamics*, vol. 26, no. 4, pp. 850–857, 2013.

[42] M. Jacobsen and U. T. Ringertz, "Performanct optimisation of flexible wings using multiple control surfaces," *International Forum on Aeroelasticity and Structural Dynamics*, vol. 3, pp. 1–15, 2009.

[43] M. Harbeck and A. Jameson, eds., *Exploring the Limits of Shock-free Transonic Airfoil Design*, 43rd Aerospace Sciences Meeting and Exhibition, 2005.

[44] D. Quagliarella and A. Vicini, "Viscous single and multicomponent airfoil design with genetic algorithms," *Finite Elements in Analysis and Design*, vol. 37, pp. 365–380, 2001.

[45] A. Marsden, M. Wang, J. D. Jr., and P. Moin, "Suppression of vortex-shedding noise via derivative-free shape optimization," *Physics of Fluids*, vol. 16, no. 10, pp. 21–27, 2004.

[46] K. Giannakoglou, D. Papadimitriou, and I. Kampolis, "Aerodynamic shape design using evolutionary algorithms and new gradient-assisted metamodels," *Computer methods in applied mechanics and engineering*, vol. 195, pp. 6312–6329, 2006.

[47] P. Liakopoulos, I. Kampolis, and K. Giannakoglou, "Grid enabled, hierarchical distributed metamodel-assisted evolutionary algorithms for aerodynamic shape optimization," *Future Generation Computer Systems*, vol. 24, pp. 701–708, 2008.

[48] J. Periaux, D. Lee, L. Gonzalez, and K. Srinivas, "Fast reconstruction of aerodynamic shapes using evolutionary algorithms and virtual nash strategies in a cfd design environment," *Journal of Computational and Applied Mathematics*, vol. 232, pp. 61–71, 2009.

[49] S. Walton et al., "Modified cuckoo search: A new gradient free optimisation algorithm," *Chaos, Solitons and Fractals*, 2011.

[50] S. Walton, O. Hassan, and K. Morgan, "Reduced order mesh optimisation using proper orthogonal decomposition and a modified cuckoo search," *International Journal for Numerical Methods in Engineering*, vol. 93, pp. 527–550, 2013.

[51] H. Salimi, D. Giveki, M. Soltanshahi, and J. Hatami, "Extended mixture of mlp experts by hybrid of conjugate gradient method and modified cuckoo search," *International Journal of Artificial Intelligence & Applications*, vol. 3, 2012.

[52] X.-S. Yang and S. Deb, eds., *Cuckoo search via Lévy flights*, Proceedings of World Congress on Nature & Biologically Inspired Computing, IEEE Publications, 2009.

[53] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, pp. 17–35, 2013.

[54] A. Natarajan and S. Subramanian, eds., *Bloom Filter Optimization using Cuckoo Search*, Proceedings of the 2012 International Conference on Computer Communication and Informatics, 2012.

[55] G. Selvi and T. Purusothaman, "Cryptanalysis of simple block ciphers using extensive heuristic attacks," *European Journal of Scientific Research*, vol. 78, no. 4, pp. 198–221, 2012.

[56] E. Speed, ed., *Evolving a Mario Agent Using Cuckoo Search and Softmax Heuristics*, 2nd International IEEE Consumer Electronics Societys Games Innovations Conference, 2010.

[57] R. A. Vazquez, ed., *Training spiking neural models using cuckoo search algorithm*, 2011 IEEE Congress on Evolutionary Computation, 2011.

[58] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 33043, pp. 549–567, 2010.

[59] G. Viswanathan, "Lévy flights and superdiffusion in the context of biological encounters and random searches," *Physics of Life Reviews*, vol. 5, pp. 133–150, 2008.

[60] M. D. McKay, R. J. Beckmann, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[61] K. A. Sørensen, "Phd thesis: A multigrid accelerated procedure for the solution of compressible fluid flows on unstructured hybrid meshes," *Swansea University*, 2001.

[62] "A one–equation turbulence model for aerodynamic flows," vol. 1, pp. 5–21, 1994.

[63] B. J. Evans, O. Hassan, J. W. Jones, K. Morgan, and L. Remaki, "Computational fluid dynamics applied to the aerodynamic design of a land–based supersonic vehicle," *Numerical Methods for Partial Differential Equations*, vol. 27, pp. 141–159, 2010.

[64] Wales, "High performance computing wales," *Online Resource: http://www.hpcwales.co.uk/*, 2014.